



Version 10 for Mac OS X
User Manual

Introduction

An introduction to SourceGuardian 10 for Mac OS X

by Team SourceGuardian

This SourceGuardian 10 for Mac OS X User Manual covers all of the features in this new exciting version. We hope that you enjoy using our product and find this user guide to be informative.

If there is anything that you feel has been omitted from this user manual, then please let us know as we are passionate about providing excellent service.

Have fun using your new product...

SourceGuardian 10 for Mac OS X

© 2001-2014 SourceGuardian Ltd.

All rights reserved. No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of the publisher.

Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. The publisher and the author make no claim to these trademarks.

While every precaution has been taken in the preparation of this document, the publisher and the author assume no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that may accompany it. In no event shall the publisher and the author be liable for any loss of profit or any other commercial damage caused or alleged to have been caused directly or indirectly by this document.

Table of Contents

Foreword	0
Part I Introduction	2
1 About SourceGuardian for PHP	2
2 How to buy	2
3 Features	2
Part II GUI manual	6
1 Overview	6
2 Registration	6
3 Welcome screen	8
4 How to start?	8
5 Project	9
Choosing files and destination	9
Locking options	12
Generating a license file	17
Advanced options	19
Encoding	23
6 Installing Loaders	24
7 Copy Loaders	26
8 Getting file information	27
9 Setting preferences	28
10 Viewing registration information	31
11 Getting help	31
12 Checking for update	31
Part III Protected script loaders	33
1 Loader installation	33
2 Automatic loading	35
3 Loader filename structure	36
4 Zend extension support	37
5 Execute only SourceGuardian protected scripts	37
Part IV Command line encoder	40
1 Command line tools installation	40
OS X	40
Linux	40
Windows	40
2 Command line encoder	40
First run	41

Usage	41
General options	42
Locking options (full version only)	43
Advanced options	47
Custom predefined constants	50
Custom error handling	51
Excluding files by the file mask	52
Excluding files from encoding but still copying to output directory	52
Encoding directory content without specifying filemasks	53
Output directory for encoded scripts	53
Encoding to standard output	54
3 Script license generator (full version)	54
Usage	55
4 File information tool (full version)	57
Usage	57

Part V Encoding of HTML templates and other non-PHP files 59

1 GUI	59
2 Command line interface	59
3 Using protected templates	60
4 Using protected templates with the Smarty template engine	61
5 Creating custom encoded files from protected scripts	62
6 Using encoding SourceGuardian API from unprotected script	63
7 Debugging of scripts which work with encoded templates	63

Part VI Common mistakes 66

1 Encoded scripts modification	66
2 Extension directory (php.ini setting)	66
3 Getting "This protected script can run only in conjunction..." error	66

Part VII Advanced users 68

1 PHP shell scripts encoding	68
2 SourceGuardian loader API	68
3 Marking a file to be skipped during encoding	69

Part VIII License 71

1 SourceGuardian License	71
2 SourceGuardian Loaders License	76
3 Other Licenses for SourceGuardian for OSX	78
RegexKit Framework	78
Sparkle Framework	79

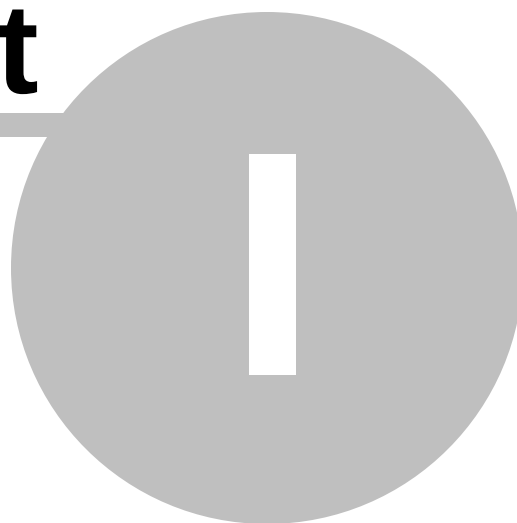
Part IX Changelog 81

1 Version 10 / June 2014	81
--------------------------------	----

2	Version 9.5 / July 2013	82
3	Version 9.0 / July 2012	83
4	Version 8.2 / April 2010	86
5	Version 8.1 / January 2010	87
	Index	0

SourceGuardian 10 for Mac OS X

Part



1 Introduction

1.1 About SourceGuardian for PHP



The SourceGuardian™ products have been built as a suite of professional systems for source code protection. Our team of programmers have created proprietary methods for encrypting code whilst keeping the maximum flexibility for the distribution of your scripts.

Our first product, SourceGuardian™ for PHP was launched in 2002 and quickly rose to become the professionals for PHP code protection. Thanks to our early market entry and the customers who put their trust in us, we've been able to develop SourceGuardian™ into a leading protection solution used by thousands across the world.

The most exciting thing about SourceGuardian™ for us is how we constantly hear from our clients how SourceGuardian™ has finally enabled them to distribute their commercial code and how developers are able to solve many of the problems that plague them when coding for a specific client. We hope to enable many more!

As for the future of SourceGuardian™, our PHP product has really taken us aback with the huge uptake and acceptance in the market and we thank everyone who has purchased, downloaded or even taken the time to browse our site. We plan to continue to increase the functionality and power of these programs whilst keeping an affordable upgrade path.

Thanks for your interest, and thanks for your business.

The SourceGuardian™ Team

1.2 How to buy

To purchase SourceGuardian™ 10 for PHP, please visit our web site:

<http://www.sourceguardian.com/purchase/index.php>

There are two methods available: via credit card or via Paypal.

1.3 Features

SourceGuardian™ 10 for PHP Features List

Protection method

The SourceGuardian™ 10 for PHP Encoder protects PHP scripts by compiling PHP source code into a bytecode format and this is followed by encryption. This protects your scripts from reverse engineering.

Supported PHP versions

SourceGuardian™ 10 for PHP works with the following versions: PHP 4.3.x, 4.4.x, 5.0 - 5.6 are fully supported.

Interface

SourceGuardian™ for Mac OS X is a universal GUI application which runs on 32 or 64-bit Intel Macintosh computers. Mac OS X 10.4 or later is required. A command line encoder is included. In addition, we also developed a powerful cross-platform GUI and command line encoders that runs under Windows and Linux.

Locking

To protect your scripts from unauthorised usage SourceGuardian™ 10 for PHP has added features that can optionally lock your scripts to run only from predefined IP addresses, domain names or LAN hardware addresses (MAC). SourceGuardian™ 10 for PHP can also easily produce trial versions of your scripts by setting an expiry date for PHP scripts or by limiting the number of days that protected script will work. To protect against local date change for trial version of your protected scripts there is an option for time checking with atomic online time servers. For larger projects SourceGuardian™ 10 for PHP provides an option to protect an entire project so that all scripts used in the project will work only with other protected scripts. No scripts may include a protected script from the unprotected script and this adds another level of protection.

Here is a sample list of features:

- locking to date with optional atomic online time servers checking
- locking to multiple domain names
- locking to multiple IP addresses
- locking to multiple LAN hardware (MAC) addresses
- improved locking to a specific domain name with encryption. The domain name is used as a part of the key for encryption, so protected scripts may not be decrypted and run from another domain.
- improved locking to the ip address with encryption. The ip address is used as a part of the key for encryption. This means that protected scripts cannot be decrypted and run from another ip address.
- locking of an entire PHP project, so that no protected script can run if any other script is substituted with an unencoded one or encoded with another installation of SourceGuardian™. This is ideal for protecting settings, passwords etc within a PHP project.
- locking to an external license file produced by the built-in SourceGuardian™ 10 for PHP license generator. This is ideal for creating protected scripts to be deployed to different users and it even allows to assign different locking options to different users. The SourceGuardian™ 10 for PHP license generator tool can run from GUI or as a command line tool which adds another powerful element - It provides a method for licenses to be dynamically generated and this would be useful (for example) when selling scripts online.
- locking protected scripts to work only online

Encoding of HTML templates and other non-PHP files

We have added an option for encoding HTML templates, or other non-PHP files, using the SourceGuardian encoder. HTML template or other non-PHP files may be encoded by the encoder, then read and decrypted from the protected scripts code. Template files which are encoded as a part of the project may be used only from protected scripts which were encoded as a part of the same project. It's impossible to use protected templates from unencoded scripts or from scripts encoded with a different SourceGuardian project.

Other options

The following is not an exhaustive list, but covers some of the other options in version 10:

- PHP 5.4, 5.5 and 5.6 support including new language features
- Improved code protection methods
- Please see the [change log](#) for further details

Cross platform

Cross platform encoding. A script encoded under one operating system will run under any other supported operating systems. Currently we have the encoder for Windows, Linux and Macintosh. Protected scripts will work on Windows, Linux, OSX, FreeBSD, OpenBSD and other OS. Script Loaders are available for these operating systems. Please find a list of all supported operating systems on [our web site](#). In the near future we will support more operating systems and can create a bespoke loader for your OS, please contact us support@sourceguardian.com if you are interested.

Thread Safety support

SourceGuardian™ 10 for PHP has a special versions of its Loaders for Thread Safety PHP installations.

Evaluation

We provide a Free 14 days evaluation of SourceGuardian™ 10 for PHP

SourceGuardian 10 for Mac OS X

Part



2 GUI manual

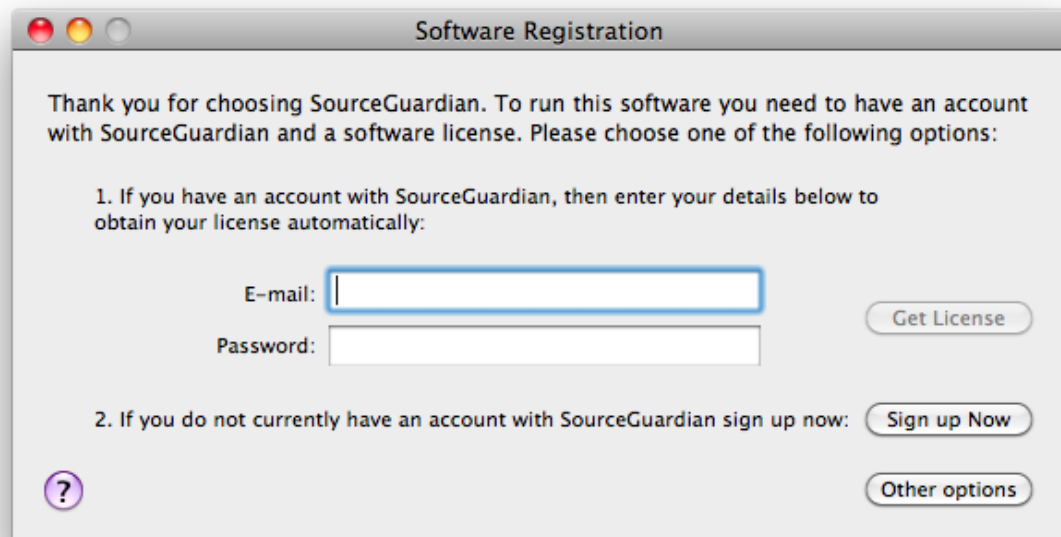
2.1 Overview



The SourceGuardian™ 10 for Mac OS X is a universal GUI application which runs on 32 or 64-bit Intel Macintosh computers, user-friendly graphic interface to our command line encoder. It uses all the powerful features that command line encoder offers, while adding many additional useful enhancements to the encoding routine. User interface is easy to use and allows access to all powerful features of the encoder. The GUI has a multi-document interface which lets you work with multiple SourceGuardian projects at the same time.

2.2 Registration

On your first run of SourceGuardian™ 10 for PHP you should see following screen:



This screen means that you need to obtain a license in order to run SourceGuardian™ . There are two ways to do this.

1. Obtaining a license from the application itself.

This is the fastest way to obtain a license, but to do it you need to have a connection to the Internet. If you don't have a connection to the Internet please use the second option ([see 2](#)) for obtaining the license using another machine connected to the Internet.

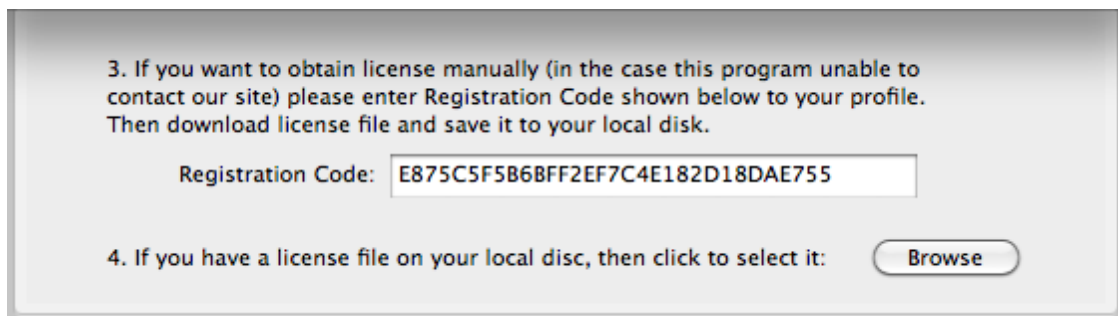
Please note, that some firewall or proxy software may prohibit SourceGuardian™ from connecting to the Internet, so you may have to enable the Internet access for SourceGuardian™ or obtain your license using another option ([see 2](#)). On how to enable the Internet access for a custom application with your firewall please consult your firewall documentation.

When you purchase a full version of SourceGuardian™ 10 for PHP, or request a demo version of it, you will receive an email with details on how to access your profile on our web site. This email contains a user name (which is your email address entered during registration) and a profile password. Please type them in the 'Email' and 'Password' fields and click on 'Get License' button. After the license has been successfully downloaded SourceGuardian™ 10 for PHP will run and you see a welcome screen. If anything has gone wrong with the installation, you will see an error message again. Make sure you have entered your email and password correctly, check your Internet settings and try again. If you cannot get a license please try another option ([see 2](#)).

2. Obtaining a license via the "your profile" section on our web site.

If you are unable to obtain a license with the online registration method above, you can use this option to retrieve and download a license file. Go to the profile login page on the SourceGuardian.com website. Type your email and password.

Click "Other options" on Software Registration screen. You will see that some additional options become visible.



3. If you want to obtain license manually (in the case this program unable to contact our site) please enter Registration Code shown below to your profile. Then download license file and save it to your local disk.

Registration Code:

4. If you have a license file on your local disc, then click to select it:

Once you have entered your user profile on the web site, select the registration code from the SourceGuardian™ 10 for PHP application (by double clicking on it). Copy and paste it to the corresponding field in the "your profile" area on our web site ("Please enter your registration key here to generate license:").

When you have done the above, click on 'Submit' in the user profile on the web site and this will generate a license. To download it click on the 'Download' link in the 'Available licenses' section. Save this license, somewhere on your local disk. After that, in the SourceGuardian™ 10 for PHP application click on 'Browse' button, select the license file you have downloaded and saved. Software Registration window will be closed on successful license registration and you will see a welcome screen.

2.3 Welcome screen



This welcome screen appears when you start SourceGuardian™ 10 for PHP and there are no project files opened yet. If there are projects opened, you may open Welcome screen in Window/Open Welcome Screen menu. The following outlines features available on this screen:

Click on "Create a New Project" if you want to start a new empty project. Default options set in [Preferences](#) will be automatically applied to a new project.

Click on "Open Existing Project" if you already have a SourceGuardian™ 10 for PHP project and want to continue working with it, run encoding or generate a script license.

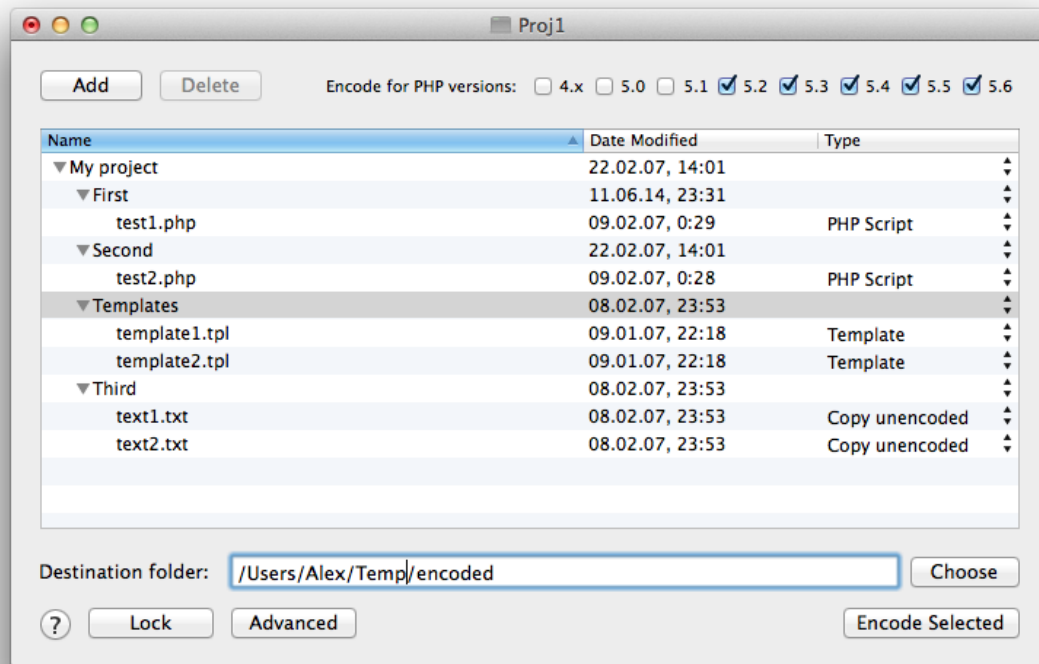
Using the top menu you can read built-in help, send support ticket to us, submit a feature suggestion and download latest loaders. See Help menu. Help is also always available by pressing Command+? shortcut.

2.4 How to start?

To start encoding PHP scripts you need to create a new SourceGuardian project, choose files for encoding, select a destination directory and click Encode. Please refer to our [Project section](#) for further details.

2.5 Project

2.5.1 Choosing files and destination



The project window is the main window you will use when working with SourceGuardian encoder. It displays a SourceGuardian project which includes information about files to be encoded, encoding mode, target PHP versions, locking and advanced options. You can save the project to a file using File/Save or File/Save As menu item. You may load a previously saved project using File/Open or choose from your recent projects using File/Open Recent menu item.

You need to create a SourceGuardian project before you can encode files. This is simple process during which you will choose files to be encoded, select destination folder, choose PHP versions, encoding mode and set encoding options. First two steps are required. All other steps are optional.

1. Choose files for encoding.

Clicking on "Add" displays a standard dialog for selecting files or folders. You may either add separate files or add entire folders. You may set encoding mode for files or delete files from the project that you do not want to encode. No real deletion happens when you delete files or folders from the project using "Remove from Project" button. Clicking on "Remove from Project" simply excludes a file or folder from encoding.

Choose encoding mode for files or folders. There are 4 modes available:

- PHP Script - the file will be encoded as PHP script
- Template - the file will be encoded as template. A file encoded as template is encrypted and may be

decrypted from your encoded PHP code using the `sg_load_file()` [SourceGuardian API](#) function. This mode is useful for encoding HTML templates, custom configuration files etc. Note, this mode differs from the "PHP script" encoding and files encoded as "Template" will not be automatically decoded by the SourceGuardian Loader. You need to use `sg_load_file()` API function to decrypt them from the protected code.

- Skip - the file will not be encoded or copied to the destination folder. It's useful for files that you do not want to encode, but want to keep in the project tree.
- Copy unencoded - the file will not be encoded and copied as is to the destination folder. It's useful for files that you want to include to the protected project but which you do not want to encode at all, e.g. text, javascript, configuration, images, audio, video etc files.

To change encoding mode for a file or folder you need to choose it from the Type dropdown in the files list. Double click to select another encoding mode. You may select encoding mode for each file separately or set it once for a folder. When you do it for a folder then selected mode will be assigned to all files and subfolders recursively.

When you add new files to the project, encoding modes are set automatically. This is done according to default settings in [Preferences](#). You may change defaults there if you often need to make changes in encoding modes after adding new files.

You may build a new "tree" of your project instead of adding existing files and folders. Use "Create Folder" from the context right-click menu to create a new folder within the project. This option is available when you are using a File Manager Mode for the project window. You may change the Project Window Mode in [Preferences](#). You may add files and folders to the newly created folders as usual. Files added to the project will be copied to the destination folder and encoded there replicating the folders structure of the project.

2. Choose a destination folder.

You need to select a destination folder where encoded files will be copied to. Files marked as "Copy unencoded" will be also copied to the destination folder as is without encoding. Click Choose button to select a destination folder. You may create a new folder by clicking "Create Folder" in the file dialog if you need.

3. Optionally set target PHP versions.

SourceGuardian encoder produces different bytecodes for different PHP versions to provide maximum compatibility and full support of PHP language features. So you need to choose target PHP versions for encoding. All versions of the internal bytecode for one source script will be packed into one protected script. Since version 8 of SourceGuardian you do not need to provide different versions of your protected code for different versions of PHP. Choose the target PHP versions by clicking checkboxes at the top of the project window. A new project will use default settings. You may change the default setting of the target PHP versions in [Preferences](#).

Your choice should be easy when you know what version of PHP is installed on the server where you plan to run your scripts. It is important to know which PHP versions your code is compatible with and do the correct choice of the target PHP versions. For example if your code uses array type hints, it will not be compatible with either PHP 4.x or PHP 5.0/5.1 and therefore you should tick PHP 5.2 and/or PHP 5.3, PHP 5.4 checkboxes. Nothing bad will happen if you do a wrong choice. A built-in PHP compiler will try to compile your code for each of the selected versions of PHP and will display an error message in the case your code is not compatible with any of the selected versions of PHP. Note, the entire script will NOT be encoded in that case and you can find details in the encoding log.

4. Optionally set locking options.

You may set many locking options which include IP address, domain name locks, hardware MAC address lock, expire date lock etc. Click Lock to set locking options. Please refer to [Locking options](#) section for further details.

5. Optionally set advanced options.

You may set advanced options which include special ASP and PHP tags support, custom header code, loader not found error handler etc. Click Advanced to set advanced options. Please refer to [Advanced options](#) section for further details.

6. Click Encode to start encoding.

If Encode button is not available then you have not added files for encoding or have not chosen the destination folder yet. Please do it to proceed with encoding. You may hold down the Command key on the keyboard before clicking the Encode to encode only selected files and folders in the list. It is useful if you need to re-encode only some files without doing it for the entire project.

2.5.2 Locking options

☒ Set expiration date 01.08.2013 ☐ Lock to a license file

☐ Script will expire in

☐ Script will only work with other encoded files ☐ Script will only work online

Run from these IP addresses only + -

IP address	Mask
127.0.0.1	255.255.255.0

Online time servers for date check + -

TIME or NTP server
pool.ntp.org

☐ Encode to IP (one IP address only) ☒ Check date with online time servers

Run from these domains only + -

Domain name

Custom defined constants + -

Name	Value
Version	9.5
Copyright	My Company

☐ Encode to domain (one domain only)

Run from these MAC addresses only + -

MAC address

Custom error handlers + -

Error condition	Function name
Restricted to run	my_handler

? Close

This popup is used for setting locking options for encoded scripts. All locking options are stored internally encoded within protected script by default. You may choose "Lock to external license file" option to put all locking settings into a special encoded file which will act as a license file for running your protected scripts. You need to specify a [license file name](#) to use this option.

Set expiration date

Choose a date you wish your script to expire. The script will not run on and after the specified date and display the following error message: "Protected script has expired".

Script will expire in (days)

Select after how many days the script will expire starting from today. The script will not run on and after the specified date and display the following error message: "Protected script has expired".

Online time servers for date check

If you use an expiration date lock option for your scripts you may wish to let them check time with online time services rather than use local server time which may be potentially changed. You may specify a list of time service servers. Old TIME and modern NTP protocols are supported. This list will be filled in for a new project using a default list from [Preferences](#).

Please note, using this option will require the Internet access on the machine where your protected scripts run. Time servers are checked following the list and if the first does not reply then the second is tried and so on. Using this option adds a delay to running your protected files because of accessing online time services via the Internet. If you use this option we suggest that you specify at least two time servers or more for reliability. If none of the specified time servers can be reached when your protected PHP code is running, it fails with an error.

Time servers will be checked for protected scripts only if "Check date with online time servers" option is also selected. This is new to version 9 of SourceGuardian. You probably need to select this option when loading an old version 8 project file in order to continue using time servers date checking.

Script will only work with other encoded files

This option makes sense only when encoding multiple files. All scripts encoded with this option will work only with other encoded files of the same project and will NOT work if any of the included files or top files are substituted with an unencoded one or encoded as a part of another project or by another installation of SourceGuardian™ for PHP. This gives you the ultimate protection for your projects when multiple PHP scripts are used together.

Example: If you have a password in a.php and then b.php includes a.php and calls c.php for any action. Enabling this option makes it impossible to substitute c.php with their own code and do 'echo \$password' to know your password. Also enabling this option makes it impossible to create d.php which includes protected a.php and then does 'echo \$password'.

Run from these IP addresses only

Lock scripts to IP/mask. The encoder will lock the script to run only from the specified IP address(es). A specified IP address mask will be applied to the real IP address before comparing. So you may use this option to lock the script to a subnet if a correct mask is specified. If a protected script is run from the IP address which is not allowed, the script terminates with the error message: "This script is not licensed to run on this machine". You may add as many IP address/mask pairs as you need. Click '+' button if you need to add another IP/Mask pair. Click '-' button if you want to delete the selected IP/Mask.

IP address mask 255.255.255.255 is used by default if not specified.

Locking to IP addresses works only for scripts which will run on web servers. As there is no definite IP address when the script runs from shell (command line) you should NOT use locking

to IP address for scripts which will run from shell e.g. for cron jobs.

Encode to IP

Lock and encrypt scripts to IP/mask. You may specify only one IP/mask. The encoder will lock the script to run only from the specified IP address. The encoder will use a specified IP address with an applied mask as a part of the key for encryption for providing maximum protection. A SourceGuardian Loader will not be able to even decrypt the script that runs from the wrong IP address and will display an error message: "Protected script checksum error". IP address mask 255.255.255.255 is used by default if not specified.

Locking to IP addresses works only for scripts which will run on web servers. As there is no definite IP address when the script runs from shell (command line) you should NOT use locking to IP address for scripts which will run from shell e.g. for cron jobs.

Run from these domains only

Lock the script to a domain name. The encoder will lock the script to run only from the specified domain and all sub domains. If an attempt is made to run the script on a non-authorized domain, the following error message will be displayed: "This script is not licensed to run on this machine". You may add as many domain names as you need.

Use the name of the main domain in this option, not the name of any subdomain until you are sure you need to lock to a subdomain.

Example 1: mydomain.com

The script will run from mydomain.com, www.mydomain.com, myname.mydomain.com etc but will NOT run from otherdomain.com, www.otherdomain.com, otherdomain.net etc.

Example 2: www.mydomain.com

Script will run ONLY from www.mydomain.com. It will not run on the main domain mydomain.com and all other subdomains like myname.mydomain.com as well as other domains like otherdomain.com, www.otherdomain.com, otherdomain.net etc.

You may use * and ? wildcards when specifying a domain name. Wildcards have their usual behavior similar to a file system.

Example 3: *.mydomain.com

The script will run from www.mydomain.com, extra.mydomain.com, myname.mydomain.com etc but will NOT run from mydomain.com, otherdomain.com, otherdomain.net etc.

Example 4: *mydomain.com (please note a change from the previous example)

The script will run from www.mydomain.com, extra.mydomain.com, myname.mydomain.com etc. AND mydomain.com but will NOT run from otherdomain.com, otherdomain.net etc.

Locking to domain names works only for scripts which will run on web servers. As there is no definite domain name when the script runs from shell (command line) you should NOT use

locking to domain name for scripts which will run from shell e.g. for cron jobs.

Encode to domain

Lock and encrypt scripts to one domain. You may specify only one domain. The encoder will lock the script to run only from the specified domain name. The encoder will use a specified domain name as a part of the key for encryption for providing maximum protection. The Loader will not be able to even decrypt the script that runs in the wrong domain name and will display an error message: "Protected script checksum error". You should not use wildcards for domain name when using this option. This option works ONLY for one strictly specified domain name.

Locking to domain names works only for scripts which will run on web servers. As there is no definite domain name when the script runs from shell (command line) you should NOT use locking to domain name for scripts which will run from shell e.g. for cron jobs.

Run from these MAC addresses only

Lock the script to LAN hardware (MAC) addresses (The "MAC" word used here has nothing about Apple Macintosh computers. MAC address is a hardware address of the local area networking LAN controller available for all platforms). This address is usually unique for each networking adapter and so it may be easily used to identify a machine. A MAC address is 6 bytes long, with each byte represented in hex and separated with a colon (:). The encoder will lock a script to run only from the machine which has a networking adapter with a specified MAC address. If there is more than one LAN adapter installed then script will check all of them. If an attempt is made to run the script on a machine that is missed a correct adapter, then the script fails with an error message: "This script is not licensed to run on this machine". You may specify multiple MAC addresses, if any one address matches then the script runs.

You may use 'ifconfig' command in Linux or OSX or 'ipconfig /all' in Windows to get a list of installed networking adapters and know MAC addresses.

Script will only work online

If you do not use time locking option you are still able to lock the scripts to work only online. Select this option if you need your protected scripts to work only online (when the Internet connection is available). SourceGuardian Loader will check if the script it is running online by trying to access one of the time servers specified in the list.

Custom defined constants

SourceGuardian let you define custom named constants during encoding process or within an external license file. Constant name/value pairs are stored internally in the encrypted area of the protected script or the external license file. They may be used for custom script locking, adding copyrights or any other actions if you need to store a custom value in protected scripts or your script license file and then retrieve it from your protected PHP code.

You may define multiple name/value pairs for your custom constants.

To get a predefined constant value from the **encoded script** use `sg_get_const()` [SourceGuardian API](#) function. This function is defined in the SourceGuardian loader.

Syntax: `string sg_get_const( string )`

Will return a predefined SourceGuardian constant value or FALSE if constant with the specified name is not defined. SourceGuardian constants names are **case sensitive**.

There are 5 constants predefined for all protected scripts:

<code>sg_get_const("encoder")</code>	Returns the name of the encoder - "SourceGuardian"
<code>sg_get_const("version")</code>	Returns version number of the encoder
<code>sg_get_const("encode_date")</code>	Returns UNIX timestamp for the date when the script was encoded
<code>sg_get_const("license_date")</code>	Returns UNIX timestamp for the date when the script license was created. It's may differ from the "encode_date" when an external script license is used
<code>sg_get_const("expire_date")</code>	Returns script expiration date as UNIX timestamp if it's defined in the script license or internally with in the script during encoding

Custom error handlers

You may add custom error handling functions which will catch script licensing errors. An error handler is a function which accepts two parameters:

`sg_error_handler( $code , $message )`

You may use any name for this function. Also you may have different functions for different script errors. The first argument is an error code. The second one contains a default error message.

The custom error handler function must be defined before a script licensing error may occur. The best place to define it is to use "custom header code" (see [Advanced options](#)) This header code is loaded before any license checking is done and so error handler will be always available if defined there. But you may also define a custom error handler function in another encoded file which is included before the script which may cause a license error. Don't put any passwords etc secret data if you use a "custom header code" for defining the error handler as this code is stored unencoded.

There are following error handler conditions defined:

Err	Code	Default message
Restricted to run	01,02,03	This script is not licensed to run on this machine...
License broken	06	A license file which is required to run this protected script is invalid...
License expired	09	This script has expired...
License not found	13	This protected script requires ... license file in order to run
Running offline	20	This protected script requires the Internet connection in order to run
All errors	-	-

"All errors" is a special value to specify one error handler function for all SourceGuardian error codes.

#ExternalLicenseFileName
External license file name

If you use "Lock to external license file" option then you need to specify a license file name that your project will use. Scripts encoded with this option will require a license file in order to run. If the name of the license file does not include any path then protected scripts will search for the license file in the protected script's folder, its parent folder and so on. So you may have one license file for the entire project located in the top project's folder. A path may be specified for a license file. It may be an absolute file system path or URL like <http://xxx/mylicense.lic> (you may use any name for the license file). If a protected script cannot find the specified license file it will display the following error message: "script requires ... file to run".

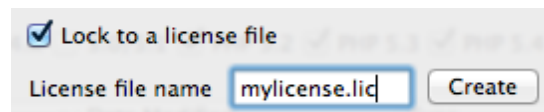
Since version 10 of SourceGuardian loaders you may use SG_LIC_PATH environment variable to specify where the loader should search for the license file. If the SG_LIC_PATH variable is set then the loader will not use any other methods of searching the license file. The environment variable should be available for PHP engine, this may require changes in the web server's configuration.

The license file may be [generated later](#) using GUI or an external license generator which is available as a command line tool for Windows, Linux and Macintosh. If you chosen to lock your scripts to an external license file this gives some additional protection to your scripts. Using locking to a license file is the best if you need to deploy one script or the entire project to different users, but need to use different restriction options for each of them.

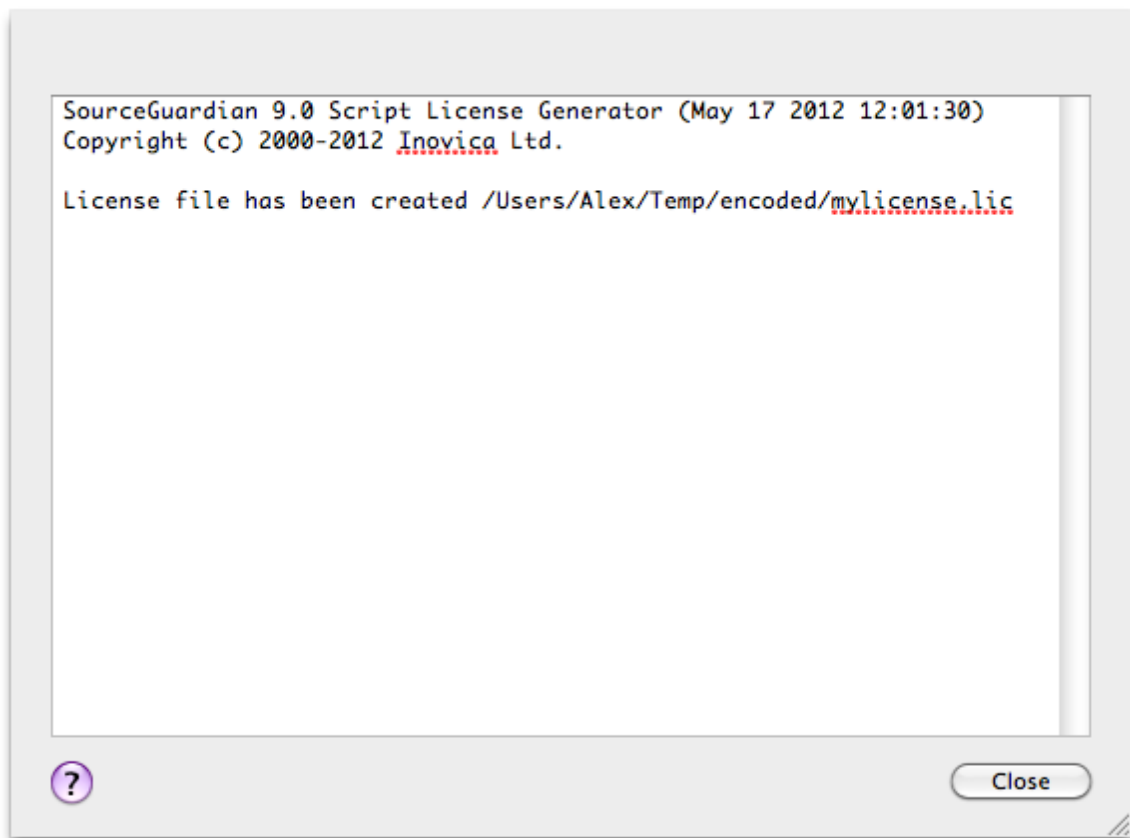
Generating a license file

To generate a license file click "Generate" button. Please refer to [Generating a license file](#) section for further details.

2.5.2.1 Generating a license file



When you select an option to lock to external license file all locking options are stored within it and this file becomes required in order to run protected script. To generate a license file you need to open locking options in the project window and click "Create" on the top next to the External license file name field. You should already have "Lock to external license file" option selected and a license file name filled in as you used this locking mode for scripts.



A new popup will display a progress in license generation. It will take a second and you will see a message saying that the license file was successfully created. **The generated license file will be saved to the project destination folder.** You may leave it there and distribute along with your protected scripts or install or send it separately to your scripts users.

Since version 9 of SourceGuardian a path may be specified for a license file. It may be an absolute file system path or URL like `http://xxx/mylicense.lic` (you may use any name for the license file). When you click "Generate" to create a license file, a specified path or URL will be ignored and the license file will be created in the destination folder. However, the specified path or URL are stored along with the required license file name in protected files when you click "Encode".

You may change locking options and generate different license files if you need. If you do so you need to copy the generated license file from the destination folder and save it separately.

You may automate the license generation using a command line license generator tool. Using the command line license generator provides a method for licenses to be dynamically created and this may be useful when selling scripts online or to automate your web site's backend etc. You may read about using the command line license generator in the ["Using external script license generator"](#) in this manual.

2.5.3 Advanced options

Project ID

Project key

☐ Enable ASP tags <% %>

☒ Enable short PHP tags <? ?>

☐ Don't integrate dynamic startup code

☐ Report E_STRICT compiler errors

☐ Report E_DEPRECATED PHP 5.3+ compiler errors

☐ Stop on compiler errors

☐ eval() compatibility mode

☐ Keep modification date for encoded files

Loader not found error code

Custom header code

License file custom text

☐ Only encode files changed since last encoding
The project was last encoded on: Never

Additional command line options

Project ID

This field lets you assign ID to your project that is used to identify what license it should accept. Specify the same Project ID in the license generator when you generate a license file for this project (if you use a command line license generator tool). This option is useful when you want to deploy several products that use external license locking so that each license works only with a corresponding Project ID.

Project Key

This field is used in conjunction with Project ID and required if you plan to use external license locking. Introduced in version 5.0 a new algorithm uses an idea of two keys. The first key (Project Id) is stored within the encrypted area of protected scripts and used to decrypt an external license file. The second key (Project Key) is stored within the license file and used to decrypt the bytecode from the protected script.

This algorithm protects your product against creating a full working copy from the demo version by some people who may be interested in this. In order to decrypt and run the protected script a valid license file for the full version of your product is required. Otherwise it's impossible to decrypt and run the bytecode. Project Id and Project Key values are required if external license protection method is selected.

Project ID and Project Key values are randomly generated for a new project. Usually you do not need to change them. If you need to use the same Project ID or Project Key value for any

reasons (e.g. when creating different projects which share the same license file) you may change the values in Advanced options manually.

Enable ASP tags

Enables use and recognition of ASP-like <% %> tags for indicating the PHP code in addition to standard <?php ?> tags.

Enable short PHP tags

Enables use and recognition of short PHP tag <? for indicating the PHP code, otherwise only standard <?php and ?> tags are recognized.

Don't integrate default loader code

You may use this option if you don't want to include the default starter code into protected scripts. Scripts encoded using this option will not be able to automatically find and load an appropriate SourceGuardian loader and you have to install the ixed loader manually to run this script. See this [section](#) about the manual ixed installation. If you already have the SourceGuardian loader installed server-wide in php.ini then this option may be useful.

Since PHP 5.2.5 dynamic extensions including SourceGuardian loaders must be installed to PHP's extension_dir folder specified in the php.ini configuration file and an appropriate extension=ixed.XX.YYY directive must be added to the php.ini in order to install the loader.

Note: if you select this option then "Loader not found error code" option has no effect (as the code is placed inside the default starter code).

Report E_STRICT compiler errors

E_STRICT "Strict Standards" warnings were introduced in PHP 5. This option instructs the encoder to warn of such messages during encoding. This option is ignored when encoding scripts in PHP 4.x mode. Usually E_STRICT warnings may be ignored but it may be a good idea to let the encoder display such warning messages and review the code. The encoder will stop if "Stop on compiler errors" option is also checked.

Note: the encoder can catch only compiler-related E_STRICT warnings. Run-time E_STRICT messages will be displayed when the protected script runs as usual according to the error_displaying option in the php.ini.

Report E_DEPRECATED PHP 5.3+ compiler errors

E_DEPRECATED warnings were introduced in PHP 5.3. This option instructs the encoder to warn of such messages during encoding. This option is ignored when encoding scripts for PHP older than 5.3. Usually E_DEPRECATED warnings may be ignored but it may be a good idea to let the encoder display such warning messages and review the code. The encoder will stop if "Stop on compiler errors" option is also checked.

Note: the encoder can catch only compiler-related E_DEPRECATED warnings. Run-time E_DEPRECATED messages will be displayed when the protected script runs as usual according to the error_displaying option in the php.ini.

Stop on compiler errors

This option instructs the encoder to stop encoding at first critical error or E_STRICT/E_DEPRECATED warning if these options are selected. This may be useful if you have many files in the project and there is a risk of missing errors and leaving some files unencoded because of it.

Note: Even if this option is off you will be able to find all error messages in the encoding log.

eval() compatibility mode

Enables eval() compatibility for encoded scripts. Normally encoded scripts cannot be run with the PHP eval() function. Additional CRC check restricts it as the protected code is passed as a string and source file is unknown. This improves security for encoded scripts that run in a standard way. However, some PHP template engines or custom code requires loading encoded PHP scripts as a string and then passing it to the eval(). In order to enable running protected scripts with eval() you may use this option and encode those files in the 'eval compatibility' mode.

Keep modification date for encoded files

This option instructs the encoder to keep the modification date for encoded files the same as modification date of source files. This may help in deploying only updated files and in some other cases of custom deployment of encoded files. The modification date for encoded files is set to the current date by default if this option is not used.

Custom header code

This options lets you add a custom header at the top of every encoded file. You may put any code to be executed BEFORE the protected scripts code. This code WILL NOT BE ENCODED (although it is still protected with CRC against changes). This may be either HTML text or PHP code. For PHP code - you should enclose your code with <?php ?> tags. This option is usually used for including copyrights into protected scripts but also it's used for including [custom error handler](#) functions.

All user {constants} that are defined in [locking options](#) will be replaced in the prepend code. Also some standard SourceGuardian constants may be used:

{SG_DATE} - current date i.e. date of encoding
{SG_LICENSEE} - SourceGuardian license owner from the SourceGuardian license file

It works in the same way also for licgen and do replacements for custom text if it is used. [See details](#)

Loader not found error code

It is possible to change the default action when an appropriate SourceGuardian loader is not installed and could not be found or used for automatic dynamic loading. The default handler included into protected script starter's code (which prepends each protected script by default) will display an error message "This script is protected by SourceGuardian™ and requires file ..." and then the script stops executing. This option allows you to change the default error action. You may use any HTML text or PHP code and it will be displayed or executed as a replacement to the default SourceGuardian™ loader error. This code WILL NOT BE ENCODED (although it is still protected with CRC against changes). This may be either HTML text or PHP code. For PHP code - you should enclose your code with `<?php ?>` tags.

License file custom text

If you use locking to an external license file you may add custom text that will be embedded as is into generated license files. The text is protected with a checksum against modification. You may include any text as user information, license description etc into license files. One text will be used for all license files unless you change it.

All user {constants} that are defined in [locking options](#) will be replaced in the text. Also some standard SourceGuardian constants may be used:

{SG_DATE} - current date i.e. date of encoding

{SG_LICENSEE} - SourceGuardian license owner from the SourceGuardian license file

It works in the same way also for the custom header in protected scripts. [See details](#)

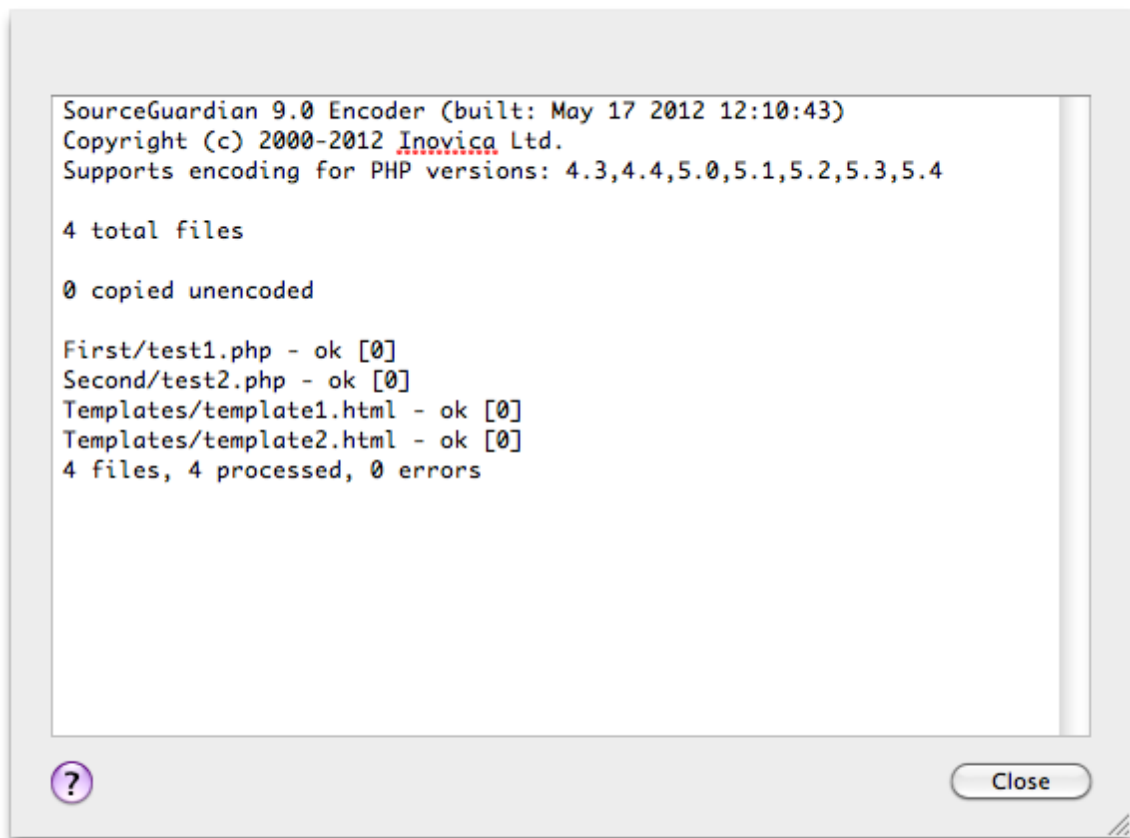
Only encode files changed since last encoding

This option lets you encode only files that have been changed since when you encoded the project for the last time. The encoder saves the date and time in the project file when the project was encoded. When this option is used the encoder checks your source files added to the project to find which files have been modified.

Additional command line options

This option lets you pass additional options to the command line encoder when it's being called for encoding of the project. Normally, you do not need to add any options as all of the options are available on the 'Lock' and 'Advanced' windows in GUI. **Use this on your own risk. Specifying unexpected options to the command line encoder may cause unexpected results or even files loss.** This option was added in version 10 of SourceGuardian for special support cases and may be removed in future releases. If there are any issues with encoding of some code or issues with running of the encoded code, our support team may suggest that you add some special options to the encoder using this line for debugging purposes.

2.5.4 Encoding



After you have added files to your project, selected a destination folder and optionally set locking and advanced options you may start encoding your scripts. This is simple - just click on "Encode" in [project window](#). A popup window will display the log of encoding. Finally you can see the number of processed files and files which could not be processed because of errors.

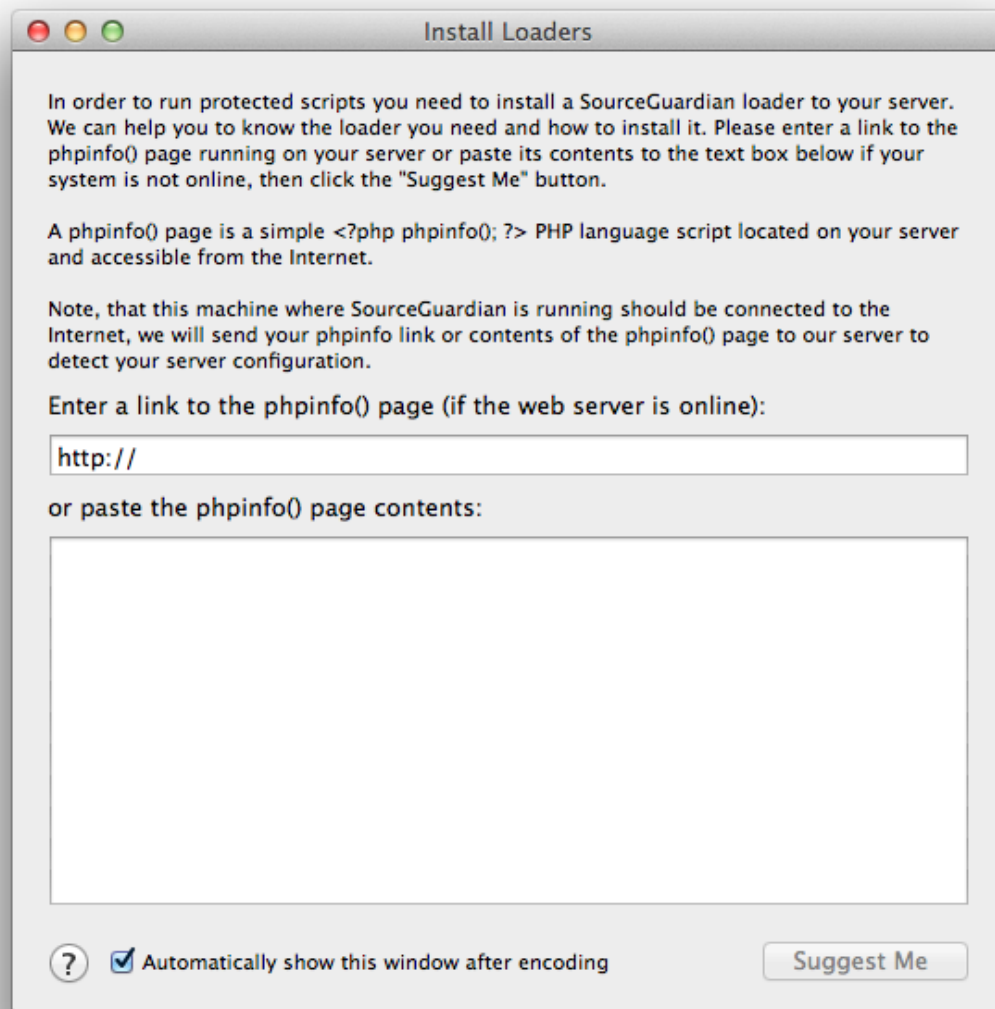
You may see the following error messages next to file names:

Error message	Description
ok	The script or template was encoded without problems.
file not found	File for encoding was not found (this error should not appear in the GUI).
PHP compiler error	You have an error in your script and the encoder could not compile it into bytecode. This message is specific to the PHP version which is also displayed as well as the line number. Check if your script is free from errors and that <u>it is compatible with the version of PHP you are encoding for</u> . E.g. using of some new language features are only possible with recent versions of PHP. Encoding such scripts for older versions may result in the error message.
could not write file	The destination folder that you selected is not writable.
file is already processed by SourceGuardian	File is already encoded with SourceGuardian and could not be encoded once again (this error should not appear in the GUI).

empty file, skipped	Empty files could not be encoded and not processed. If you need to have this empty file in the destination folder, change encoding mode to "Copy unencoded" for it.
not regular file, skipped	The file you have selected for encoding is not a regular file (for example it's a device driver file etc)
copied	This is normal. You have selected "Copy unencoded" for this file and it was copied to destination folder.
internal encoder error	Please let us know about this error. Send us a support ticket including the file that caused this error.
unknown error	Please let us know about this error. Send us a support ticket including the file that caused this error.

You may stop encoding by clicking on "Cancel". Once encoding is finished click on "Close" to close the encoding log window and return to your project.

2.6 Installing Loaders



We have created this "Install Loaders" wizard in order to help you know the loader you need and how to install it to your target system where protected PHP files will run. Please read the text about SourceGuardian loaders on this screen. This wizard checks your remote system and know how to install the loader to it. You need to create a simple phpinfo page on your remote machine. The phpinfo page is a code like this (without quotes) "<?php phpinfo(); ?>" which is located on your remote system and available from the Internet. Please enter a HTTP link to access the phpinfo page on your remote machine and click "Suggest Me" button to get instructions of how to download and install an appropriate loader to your remote machine.

If you are running a web server locally on the same machine which runs SourceGuardian you may enter a local HTTP link like this `http://localhost/path/to/phpinfo.php` (specifying a real path to the phpinfo() page running in your local web server of course).

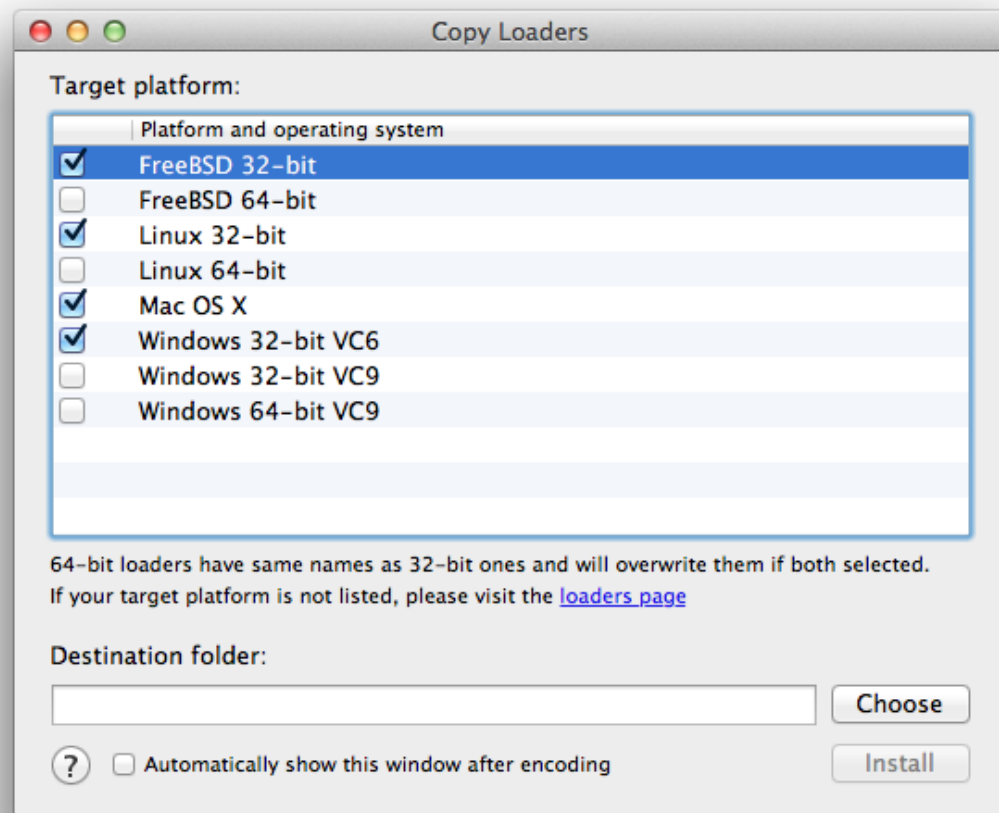
If your target machine is not available online you may copy the phpinfo() page contents from it and paste

it to the large text box in this window. You may copy either HTML or plain text contents of the phpinfo() page. Then click "Suggest Me" to know details about how to download and install an appropriate loader for your system.

To display this "Install Loaders" window you may select File/Install Loaders menu item.

This window is automatically shown when the project has been successfully encoded. You may deselect a checkbox if you do not want this to happen every time you encode your project. You may restore this option in [Preferences](#) after it has been switched off.

2.7 Copy Loaders



Loaders for some operating systems are prepackaged with the encoder. To copy loaders into encoded scripts folder select File/Copy Loaders menu item. Select loaders to install, choose a destination folder and click Install. Loaders will be copied to the /ixed/ subfolder within the destination folder.

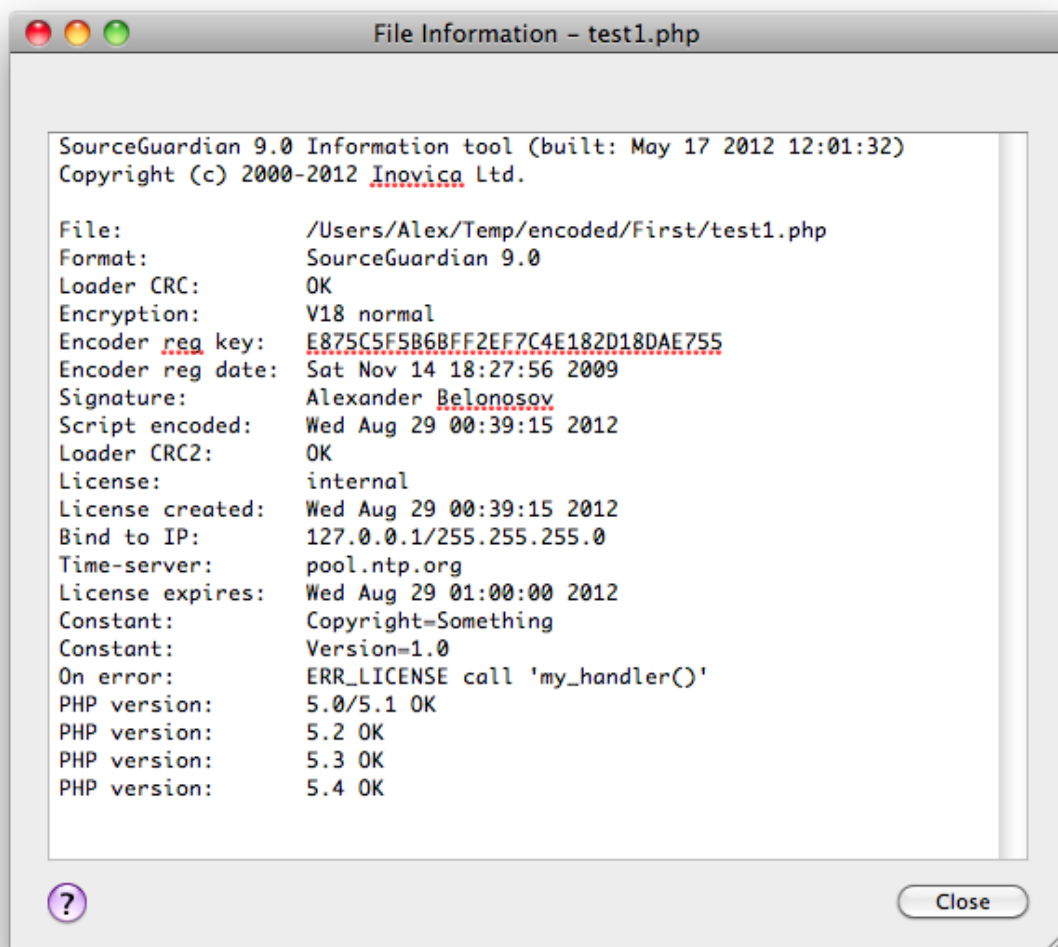
In order to make the installation package smaller we do not include loaders for all operating systems. We support more operating systems than you can see in the Copy Loaders window.

Please visit our [loaders page](#) if your operating system is not listed or to download the latest version of the loader.

Please note: 64-bit loaders have the same names as 32-bit ones and will overwrite them if both selected. VC6 and VC9 loaders for Windows also have the same names and ones will overwrite the others if both VC6 and VC9 options have been selected.

This window is automatically shown when the project has been successfully encoded. You may deselect a checkbox if you do not want this to happen every time you encode your project. You may restore this option in [Preferences](#) after it has been switched off.

2.8 Getting file information



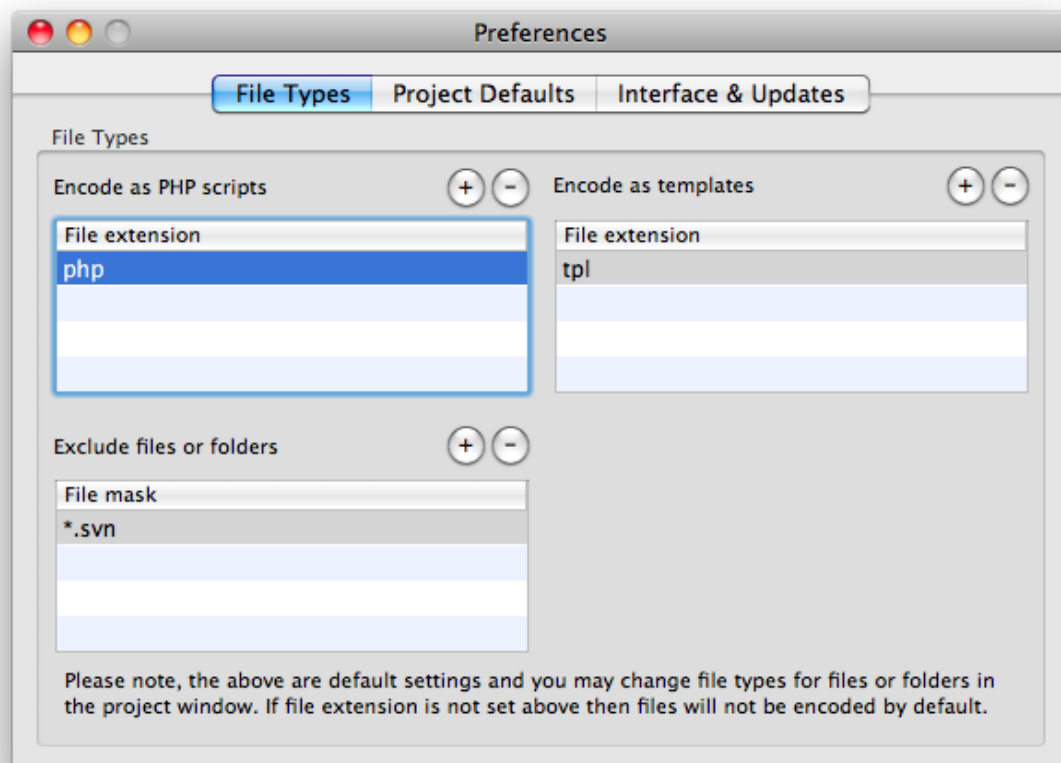
You may get information about a protected script or a license file. This may be useful for supporting your customers, checking scripts or licenses passed to them etc. You may know the date of encoding,

expiration date, binding options etc parameters from the protected script or the script license.

Choose File Information from the File menu, then select a protected file or a license file from the dialog. Information about the protected script or the external license will be displayed.

It's possible to display script information only for files created **with the same installation** of SourceGuardian. If the script is locked to an external license file and the license file is also available then the information about the license will be included to the output. As Project ID and Project Key values are both required for getting the license information you may get this information only if you open the project that the license was generated from, and use File Information then. Project ID and Project Key values from the current opened project will be used for decoding the license information.

2.9 Setting preferences



Choose Preferences from the File menu to display Preferences window.

File Types

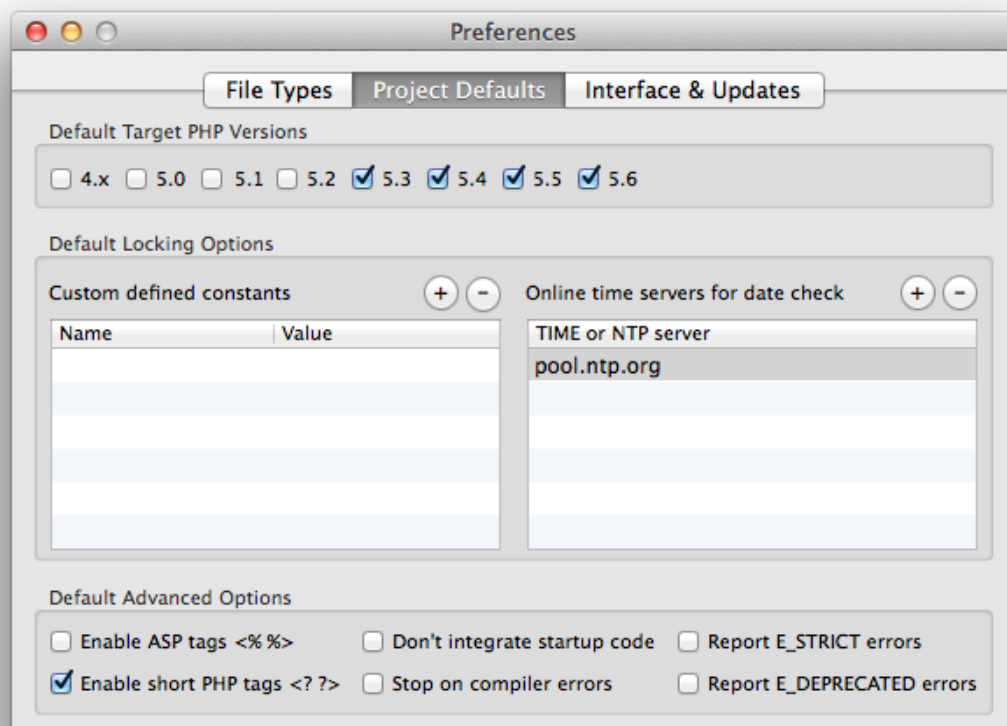
File types tab allows you to set file types which will be used for setting default encoding mode when you add files to the project. Also you can specify file types which will not be added to a project when you add multiple files or folders. Use "Encode as PHP scripts" list to set file types for encoding using PHP

script mode by default. Use "Encode as templates" list to set file types for encoding using "Template" mode by default (see the [Project section](#) in this manual for further information about possible encoding modes). Please add only file extensions without a period in the lists.

The "Exclude list" lets you specify files or folders that should not be added into the project when you add files or folders. When specifying files or folders to be excluded you may specify exact file names or use file masks (* and ? symbols may be used)

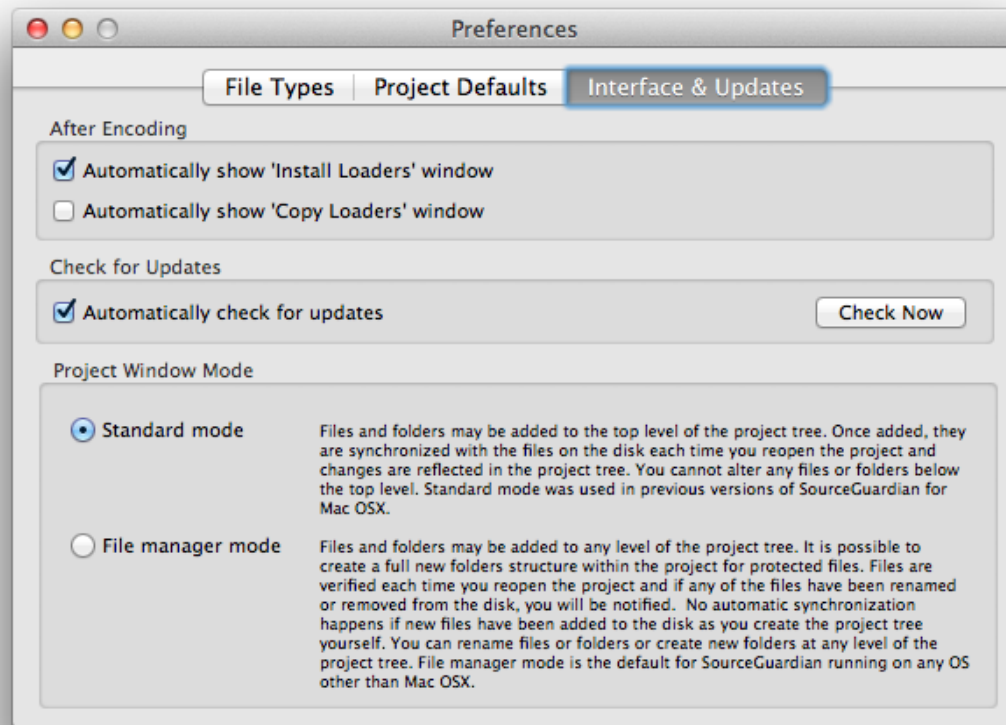
Default setting is to encode php, inc, htm and html files as PHP scripts and tpl files as HTML templates.

Please note, the above lists set only default behavior. You may always change encoding mode per file or folder in the [project window](#).



Project Defaults tab

Project defaults tab allows you to set default encoding options which will be used for new projects. You may set default target PHP versions, define custom constants, define a list of online time servers you prefer to use for date checking, set default [advanced](#) options. This tab is useful for setting up default options to values you usually need for all your projects.



Interface & Updates

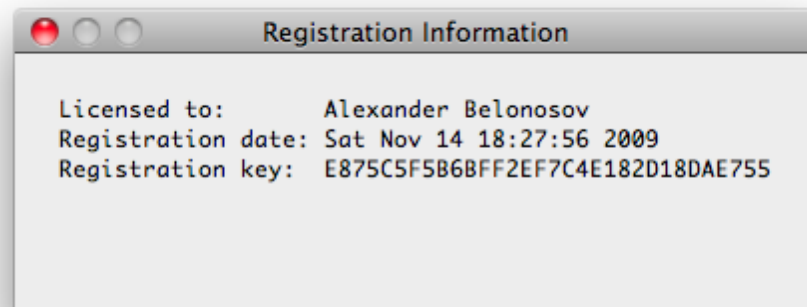
Use Interface & Updates tab to setup if "Copy Loaders" window and "Install Loaders" window will be shown after successful encoding. Use "Check for updates" option to enable or disable automatic checking for new versions of SourceGuardian. Click on "Check Now" to check for the update manually.

Project Window Mode

Since version 9 of SourceGuardian you may select how the project window will work when you add new files and folder and when you reopen an existing project file. Two options are available:

- **Standard Mode.** Files and folders may be added to the top level of the project tree. Once added, they are synchronized with the files on the disk each time you reopen the project and changes are reflected in the project tree. You cannot alter any files or folders below the top level. Standard mode was used in previous versions of SourceGuardian for Mac OSX.
- **File Manager Mode.** Files and folders may be added to any level of the project tree. It is possible to create a full new folders structure within the project for protected files. Files are verified each time you reopen the project and if any of the files have been renamed or removed from the disk, you will be notified. No automatic synchronization happens if new files have been added to the disk as you create the project tree yourself. You can rename files or folders or create new folders at any level of the project tree. File manager mode is the default for SourceGuardian running on any OS other than Mac OSX.

2.10 Viewing registration information



You may check your license information by choosing Registration Information from the SourceGuardian Application menu. Your name, registration date and support end date will be displayed. For evaluation version it also includes the expiry date.

2.11 Getting help

Using the Help menu you can read built-in help, send a support ticket to our support team, download latest loaders and access our web site. Help is also always available by pressing Command+? shortcut. Most windows within the application include a context help available by clicking on the ? (question mark) button.

If you have any questions about using SourceGuardian and could not find the answer in our manual or have any suggestions for our product feel free to contact us support@sourceguardian.com

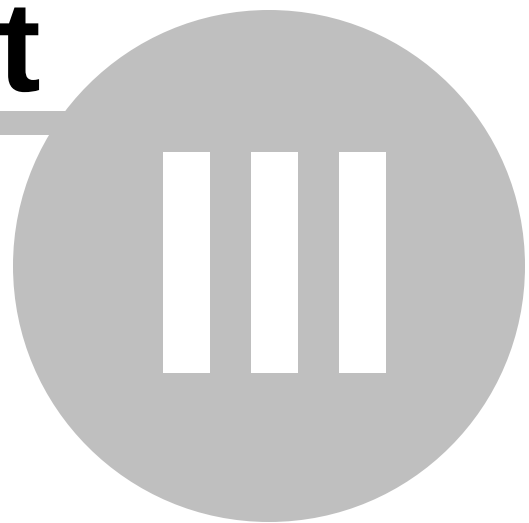
2.12 Checking for update

We keep working on SourceGuardian for PHP and release updates periodically. SourceGuardian does automatic updates by default and will inform you when a new version of SourceGuardian is available. You may also check for updates by click on "Check Now" in [Preferences](#). Automatic updates may be switched off in Preferences but we recommend you to leave this option on to keep your SourceGuardian installation up to date.

We update loaders when new official version of PHP is released. To check for loaders update please visit the loaders page on our web site by selecting Help/Download Latest Loaders or click to open this link in your browser <http://www.sourceguardian.com/loaders/>

SourceGuardian 10 for Mac OS X

Part



3 Protected script loaders

Scripts protected with SourceGuardian™ require the installation of a SourceGuardian™ loader on the target machine in order to run. Protected script loaders are dynamically loaded PHP extensions which load the protected script, decrypt it and then run the bytecode. The source code is never restored at any time, even in memory. There are different versions of the loaders available for different operating systems and PHP versions. Protected scripts will automatically attempt to find the loader in the `ixed/` subdirectory located within the protected script's directory or parent directories. SourceGuardian™ loaders may also be installed into PHP's `extension_dir` folder and the `php.ini` configuration file. This is the only way to run protected files if automatic loading is not supported by your OS and PHP or if faster performance is required. We recommend that you install the loader to PHP's `extension_dir` and `php.ini` even if dynamic automatic loading is possible.

For PHP versions 5.2.5+, SourceGuardian™ loaders need to be installed into the PHP extensions directory (`extension_dir`). You may find the `extension_dir` path in the `php.ini` configuration file or in the `phpinfo()` output. A way the dynamic loading `dl()` function works in PHP has been changed since version 5.2.5 - It may load PHP extensions located ONLY in the `extension_dir` directory or a subdirectory within it. This means that SourceGuardian™ loaders cannot be loaded automatically from the `ixed/` directory located within the protected script's directory or parent directories for PHP 5.2.5+. Usually you will get the following error message in that case: "Warning: `dl()` [function.dl]: Temporary module name should contain only filename". Please also read the note below about installing the loader for PHP 5.2.5+.

We periodically update SourceGuardian™ Loaders. The latest loaders are always freely available from <http://www.sourceguardian.com/loaders/>

3.1 Loader installation

SourceGuardian loader should be installed into the PHP `extension_dir` and `php.ini` for PHP 5.2.5+ due to limits in PHP engine. The reason is that SourceGuardian loader is a PHP extension and it's true for any other PHP extension for PHP since 5.2.5.

We recommend that you use a built-in loaders helper from File/Install Loaders menu. Please read [this section](#) in the user manual to know how to use this option.

Alternatively you may use our online loader assistant to know the loader you need and how to install it to your target system. The loader assistant is always available from our web site <http://sourceguardian.com/loaders/download.php>

If you still want to install the loader manually please follow instructions below.

Although it's possible to use automatic loading for old PHP versions in some conditions, if you use PHP < 5.2.5 you still need to install the loader manually if:

1) Operating system and PHP mode:

Linux, FreeBSD, OpenBSD, MacOSX, other UNIX - PHP installed as a webserver's module (with thread safety on)

Windows - PHP installed as webserver's module (thread safety is always on)

2) If Thread Safety is enabled. You may check `phpinfo()` output for this. PHP installed as a webserver's module under Windows always has Thread Safety on.

- 3) If dl() is disabled. You have "enable_dl=Off" setting in the php.ini configuration file.
- 4) If safe_mode is on.

Usually installation requires permissions to access the extension_dir directory and the php.ini configuration file. Manual installation may be used even if automatic loading is available. Having SourceGuardian™ loader installed to extension_dir and php.ini you give the maximum performance for your protected scripts. A reason is the script does not need to search for the loader every time it runs.

To install a SourceGuardian™ loader you need to do the following:

- 1) Choose an appropriate loader for your operating system and version of PHP. Please refer to the "Loader filename structure" section below to know which loader is required for your operating system and version of PHP.**
- 2) Find the loader file in the /Loaders subdirectory within SourceGuardian main installation directory and copy it to the PHP extension directory (extension_dir - check the phpinfo() output). We also suggest that you check our site for an updated version of loaders <http://www.sourceguardian.com/loaders/>**
- 3) Find the location of the php.ini configuration file (check the phpinfo() output) and add "extension=ixed.X.X.YYY" directive at the end of the file (X.X is the major version of PHP and YYY is the name of operating system). This will depend on your OS, PHP version, Thread Safety mode. Please refer to the "Loader filename structure" section below.**
- 4) Restart the webserver in order to apply changes done in the php.ini configuration file and reload PHP.**
- 5) Optionally you may open the phpinfo() page now to check that SourceGuardian loader has been successfully installed - search for "SourceGuardian" string.

Important information for Windows as a target platform for running protected files.

- a) If PHP engine is installed as a dynamic library and loaded automatically during web server start then the PHP engine is probably compiled with thread safety ON. It means that you need to use "ts" version of the loader. If you use all-in-one package like WAMP or XAMPP then the included PHP engine is also compiled with thread safety ON and you need to use "ts" version of the loader in that case, e.g. ixed.5.3ts.win for PHP 5.3.x.
- b) There are VC6 and VC9 version of loaders for Windows. VC6 versions are compiled with the legacy Visual Studio 6 compiler. VC9 versions are compiled with the Visual Studio 2008 compiler. If you are using PHP with Apache1 or Apache2 from apache.org you need to use the VC6 versions of PHP and the loader. If you are using PHP with IIS or third-party builds of Apache you should use the VC9 versions of PHP and the loader. Usually you do not use VC9 version with apache.org binaries.
- c) There are loaders for 64-bit version of Windows for PHP 5.3+ VC9. Earlier versions of PHP have no officially supported sources for 64-bit Windows and cannot be compiled in this environment. As a result SourceGuardian has support for Windows 64-bit loaders for PHP 5.3+. Only VC9 versions are supported for 64-bit Windows.

Examples of SourceGuardian loader names:

extension=ixed.4.3.lin	# for Linux, non thread safe, PHP 4.3.x
extension=ixed.5.0.0.lin	# for Linux, non thread safe, PHP 5.0.0
extension=ixed.5.0.1.lin	# for Linux, non thread safe, PHP 5.0.1
extension=ixed.5.0.2.lin	# for Linux, non thread safe, PHP 5.0.2
extension=ixed.5.0.lin	# for Linux, non thread safe, PHP 5.0.3+
extension=ixed.5.1.lin	# for Linux, non thread safe, PHP 5.1.x
extension=ixed.5.2.lin	# for Linux, non thread safe, PHP 5.2.x
extension=ixed.5.3.lin	# for Linux, non thread safe, PHP 5.3.x
extension=ixed.5.0ts.lin	# for Linux, thread safe, PHP 5.0.3+
extension=ixed.4.3.fre	# for FreeBSD, non thread safe, PHP 4.3.x
extension=ixed.4.3ts.win	# for Windows, thread safe, PHP 4.3.x
extension=ixed.5.2ts.win	# for Windows, thread safe, PHP 5.2.x
extension=ixed.5.3ts.win	# for Windows, thread safe, PHP 5.3.x

3.2 Automatic loading

If you use PHP < 5.2.5 there is a chance you can use automatic loading. It means that protected scripts will be able to find an appropriate loader automatically and use it. However, if you have permissions to install the loader server-wide to extension_dir and php.ini we strongly recommend that you do it. Server-wide installation of the loader increases performance of the protected scripts.

A protected script will be able to find and load an appropriate loader if:

1) Operating system and PHP mode:

Linux, FreeBSD, OpenBSD, MacOSX, other UNIX - PHP is installed as CGI or CLI

Linux, FreeBSD, OpenBSD, MacOSX, other UNIX - PHP is installed as a webserver's module (with thread safety off)

Windows - PHP is installed as CGI or CLI

2) Thread Safety is disabled. You may check phpinfo() output for this.

3) dl() is enabled. You should have enable_dl=On in your php.ini.

4) The PHP extensions directory (extension_dir) needs to exist. Please check that the extension_dir= option in php.ini points to the real directory. Some hosting companies have incorrect installations of PHP and this can cause problems.

5) The latest loaders are installed to the ixed/ subdirectory within your scripts directory or any parent directory.

6) **PHP version is older than 5.2.5.**

7) safe_mode is off.

8) (For Windows) extension_dir= option in php.ini should point to the directory located **on the same drive** with your document root and scripts directory.

Please note: if your server and PHP configuration conform to all conditions above for automatic loading except only a PHP version, then it is enough to copy an appropriate loader to the PHP extension directory (extension_dir). The loader will be used automatically from the extension_dir directory - no need for changes in the php.ini configuration file.

Example 1:

(loaders are in the /ixed/ subdirectory within the scripts directory)

/home/mysite/www/myscript1.php	- your protected script(s)
/home/mysite/www/myscript2.php	- your protected script(s)
/home/mysite/www/subdir/otherscript1.php	- other protected script(s)
/home/mysite/www/subdir/otherscript2.php	- other protected script(s)
/home/mysite/www/ixed/ixed.*	- SourceGuardian loaders

Example 2:

(loaders are in the /ixed/ subdirectory within a parent directory)

/home/mysite/www/myscript1.php	- your protected script(s)
/home/mysite/www/myscript2.php	- your protected script(s)
/home/mysite/www/subdir/otherscript1.php	- other protected script(s)
/home/mysite/www/subdir/otherscript2.php	- other protected script(s)
/home/ixed/ixed.*	- SourceGuardian loaders

3.3 Loader filename structure

The following provides an overview of the SourceGuardian loader naming conventions:

ixed.X.Y.Zdd.os

- X.Y - major PHP version number (4.3 for 4.3.x, 5.0 for 5.0.x)
- Z - minor PHP version number (2 for 5.0.2)
- This part may be missed in the loader name which means that this loader is for all higher PHP versions, e.g.:
 - ixed.4.3.lin - for all PHP 4.3.x versions
 - ixed.5.0.0.lin - for PHP 5.0.0 only
 - ixed.5.0.1.lin - for PHP 5.0.1 only
 - ixed.5.0.2.lin - for PHP 5.0.2 only
 - ixed.5.0.lin - for all PHP 5.0.3+ versions and higher
 - ixed.5.1.lin - for all PHP 5.1.x
 - ixed.5.2.lin - for all PHP 5.2.x
 - ixed.5.3.lin - for all PHP 5.3.x
- dd - optional code
 - (if missed) - this is the loader for non-thread safe version of PHP. Most UNIX installations are non-thread safe.
 - ts - this is the loader for thread safe version of PHP. Most Windows installations are thread safe, most Unix installations are not.
- os - three char code of operating system type.
 - .win - Microsoft Windows
 - .lin - Linux (32 and 64 bit versions available)

- .fre - FreeBSD (32 and 64 bit versions available)
- .ope - OpenBSD (32 and 64 bit versions available)
- .dar - OS X (universal binary version)

For some operating systems there are different versions of loaders for 32-bit and 64-bit mode. File names of such loaders are the same as it is impossible to determine 32/64-bit mode at the PHP level. However, 32-bit and 64-bit loaders are packed in different zip (tar.gz, tar.bz2) files so you can easily distinguish them. You need to use a correct 32-bit or 64-bit version of the loader for your system according to the platform and options PHP executable or shared object is built. You may safely try 32-bit version and then 64-bit one if you are unsure. Usually, you will get the following error message in the case of a wrong 32/64-bit loader is installed: "Unable to load dynamic library 'ixed....' cannot open shared object file" or "Unable to load dynamic library 'ixed...' wrong ELF class: ELFCLASS32(64)". If you have access to a command line shell you may check if your PHP is 32-bit or 64-bit using the command line "file" tool, e.g. "file /path/to/php".

Note, some operating systems such as Windows or Linux may no problem run 32-bit executable including PHP even if the OS itself is 64-bit. So, you may have 32-bit PHP installed and running on a 64-bit OS. This may cause misunderstanding, please check if your PHP is 32 bit or 64 bit, not the OS. However, if your are running 32-bit OS then PHP can definitely be only 32-bit too.

3.4 Zend extension support

SourceGuardian loaders may be loaded as Zend extensions. This lets you specify a full absolute path to the loader regardless of the `extension_dir` setting. Of course the PHP or webserver process should have enough permissions to access the loader in that location.

To install the loader for non thread-safe PHP use `zend_extension` option in `php.ini`:

```
zend_extension = /usr/local/ixed/ixed.4.3.lin
```

For thread-safe PHP use `zend_extension_ts` option in `php.ini`: (mod_php apache module is always thread safe in Windows)

```
zend_extension_ts = /usr/local/ixed/ixed.4.3ts.lin
```

You need to specify an appropriate loader for your OS and PHP version. See [loaders filename structure](#) section.

3.5 Execute only SourceGuardian protected scripts

It's possible to setup PHP to execute only SourceGuardian protected scripts. The SourceGuardian loader should be installed **server-wide** in `php.ini` and then the following option must be set in the `php.ini`:

```
[SourceGuardian]
sourceguardian.restrict_unencoded = "1"
```

If any unencoded script is executed the following error message appears:

Fatal error: SourceGuardian Loader - unencoded script cannot be executed [08]

SourceGuardian 10 for Mac OS X

Part

IV

4 Command line encoder

4.1 Command line tools installation

4.1.1 OS X

SourceGuardian command line encoder and tools are already installed if you have installed SourceGuardian™ 10 for PHP in OS X. The command line encoder and tools are located within the SourceGuardian application bundle.

SourceGuardian.app/Contents/MacOS/

A SourceGuardian command line encoder executable is named *sgencoder*

4.1.2 Linux

SourceGuardian command line encoder and tools are already installed if you have installed SourceGuardian™ 10 for PHP in Linux. The command line encoder and tools are located within the SourceGuardian installation directory.

A default path for the full version installation is:
<your home directory>/SourceGuardian

A default path for the demo version installation is:
<your home directory>/SourceGuardian-Evaluation

A SourceGuardian command line encoder executable is named *sgencoder*

4.1.3 Windows

The command line encoder is already installed if you have installed SourceGuardian™ 10 for PHP in Windows. SourceGuardian command line encoder and tools are located in the SourceGuardian™ 10 for PHP installation directory.

A default path for the full version installation is:
C:\Program Files\SourceGuardian 10

A default path for the demo version installation is:
C:\Program Files\SourceGuardian\SourceGuardian 10 Evaluation

A SourceGuardian command line encoder executable is named *sgencoder.exe*

4.2 Command line encoder

SourceGuardian™ 10 for PHP command line encoder and tools are available along with GUI application for Windows, OS X, Linux. You need have an access to a terminal or any kind of remote shell in order to use it. Although SourceGuardian includes GUI application you may prefer to use a command line encoder for some reasons, for example for automatic encoding or license generation in your web site's backend.

SourceGuardian™ command line encoder executable is named *sgencoder* for OS X and Linux, *sgencoder.exe* for Windows. The command line encoder may be found in the GUI installation folder (Windows and Linux) or within the application bundle /Applications/SourceGuardian.app/Contents/

MacOS (Mac OS X). Command line encoder and tools may also be installed separately.

4.2.1 First run

If you already have a GUI application installed it automatically registers your copy of SourceGuardian™ on the first run and so you do not need to follow manual installation steps below. You may already start using the command line encoder and tools.

Installing a license for the command line encoder

On the first run you have to read and accept a SourceGuardian™ 10 for PHP license agreement. Please read it and, if you accept the agreement, press Enter/Return key for the next page. You need to accept the all the terms to continue installation and run SourceGuardian.

After accepting the SourceGuardian™ 10 for PHP license you will get a web link to our site and a hexadecimal registration code on the screen. Please open the following link in your browser <http://www.sourceguardian.com/profile/> to access your user profile. Login using your registration email and the password we sent in the email, enter the hexadecimal registration code in your user profile page to register your copy of SourceGuardian and generate a license. Download a license file (encode.lic) and copy it **to the command line encoder installation directory**.

4.2.2 Usage

Running the command line encoder included in GUI

If you are running the command line encoder installed with GUI, please always specify a full path to the SourceGuardian license file (encode.lic) using the -L /path/to/encode.lic command line option. It's not needed if you installed the command line encoder separately and the license file was created in the command line encoder's installation folder.

The encode.lic license file is usually located in the following folder:

Windows XP: C:\Documents and Settings\USER\Application Data\SourceGuardian\encode.lic

Windows 7+: C:\Users\USER\Application Data\SourceGuardian\encode.lic

Example: running a command line encoder on Windows XP after installing and registering a GUI application:

```
>"C:\Program Files\SourceGuardian\sgencoder.exe" -L"C:\Documents and Settings\USER\Application Data\SourceGuardian\encode.lic"
```

Example: running a command line encoder on Windows 7 after installing and registering a GUI application:

```
>"C:\Program Files\SourceGuardian\sgencoder.exe" -L"C:\Users\USER\Application Data\SourceGuardian\encode.lic"
```

Linux: ~/.config/SourceGuardian/encode.lic

Example: running a command line encoder on Linux after installing and registering a GUI application:

```
>~/SourceGuardian/sgencoder -L~/.config/SourceGuardian/encode.lic
```

Mac OS X: /Applications/SourceGuardian.app/Contents/MacOS/encode.lic

Usually you do not need to specify a path to the encode.lic license file when running the command line encoder on Mac OS X as the license file is installed to the command line encoder's folder within the application bundle.

```
>/Applications/SourceGuardian.app/Contents/MacOS/sencoder
```

Specifying files to encode

```
single file:  sencoder -L /path/to/encode.lic [options] file.php
multiple files: sencoder -L /path/to/encode.lic [options] file1.php file2.php file3.php
file mask:    sencoder -L /path/to/encode.lic [options] *.php
file list:    sencoder -L /path/to/encode.lic [options] @filelist
```

You may run the SourceGuardian™ 10 for PHP encoder to encode either one or multiple files. You may enumerate all files you want to encode or use a file mask or file list to specify multiple files. A file list is a text file with either full or relative file paths of all the files to encode, separated by a new line (file masks are supported and you may use '*' and '?' wildcard characters). If you use a file list its name must be prepended with @ sign in the command line.

A file list passed to the SourceGuardian command line encoder for batch processing files may contain file masks. Standard wildcard ? and * symbols are supported.

An encoded file will replace the original file. The original file will be backed up (unless you turn off the backup option with a -b- option). ".bak" extension will be used by default for backups. We do not recommend that you turn off backups, make sure you have a copy of your source files!

4.2.3 General options

Available options are:

```
--phpversion <version x.y>  Encode for PHP version x.y. Possible values:
                             --phpversion 4 - encode for PHP 4.x (4.3/4.4)
                             --phpversion 5.0 - encode for PHP 5.0/5.1
                             --phpversion 5.1 - encode for PHP 5.0/5.1
                             --phpversion 5.2 - encode for PHP 5.2
                             --phpversion 5.3 - encode for PHP 5.3
                             --phpversion 5.4 - encode for PHP 5.4
                             --phpversion 5.5 - encode for PHP 5.5
                             --phpversion 5.6 - encode for PHP 5.6
                             --phpversion 5 - encode for PHP 5.x (currently supported PHP 5.0-5.6)
                             You can specify multiple --phpversion options in any combination.
```

Your code must be compatible with ALL specified versions of PHP. Otherwise you will get an error message when encoding incompatible files and such files will remain unencoded.

- v Display version number
- h Display help with full options list
- l Display license information
- w Wait for key press before exit
- q Display settings and request confirmation. Encoder will display all encoding parameters and wait for a key press before real encoding takes place. You may check all parameters and cancel if anything is not correct.
- r Recurse subdirectories. The encoder will process all subdirectories recursively when searching files using specified file masks.
- b Set an extension for backup files (bak is default).
Example: -b old
- b- Disable backup of source files (Be careful! Make sure you have a copy of your source files!)

4.2.4 Locking options (full version only)

--expire [dd/mm/yyyy]

This option lets you set an expiration date for the script. The script will not run on and after the specified date and display the following error message: "This script has expired". This option will override any previous locking settings done with the --days option.

--days [nn]

This option lets you set an expiration date in days since today. The script will not run after nn days from today and display the following error message: "This script has expired". This option will override any previous locking settings done with the --expire option.

--time-server <server,server,...>

Using atomic clock servers for expiration date checking

If you use a time lock option for your scripts you may wish to let the script check time with online time service rather than using local time on the server where your protected script is running. You may specify a list of time services during encoding.

Use --time-server option to specify time servers. You may specify multiple servers IP addresses or domain names separated with "," or ";;"

Protocols supported for time checking:

- "time" protocol (tcp to port 37)
- "NTP" protocol (udp to port 123)

If you are using a time-server option when encoding your files, they will *require* the Internet connection in order to run. Time servers will be checked in the specified order. If all servers from the specified list are offline when the protected script is running then an error message will be displayed and the script terminates:

"This protected script requires the Internet connection in order to run [20]"

It's a good idea to specify 2-3 time servers which lets your script work even if some of the time servers

are temporary down.

If you have multiple scripts included from each other and some of them were encoded with the time-server option then the script will access the time server only once for the first script for better performance and will use the time value from the time-server for other included scripts.

If you wish to use this option but do not know what time server to use, we suggest that you find further details at <http://ntp.org>

You may use pool.ntp.org as a time server, please read further about it at <http://www.pool.ntp.org/en/>

Locking the script to work only online

You may also use the time-server option to lock your script to run only online. Use time-server option as usual for this but don't specify an expiration date for the script. The script will try to access the online time service and will fail if it's not available (if there is no the Internet connection).

--domain [domain]

Lock the script to a domain name. The encoder will lock the script to run only from the specified domain and all sub domains. If an attempt is made to run the script on a non-authorized domain, the following error message will be displayed: "This script is not licensed to run on this machine". You may use this option more than once to specify multiple domains. This option may not be used with the --domain-encrypt option at the same time.

Use the name of the main domain in this option, not the name of any subdomain until you are sure you need to lock to a subdomain.

Example 1: mydomain.com

The script will run from mydomain.com, www.mydomain.com, myname.mydomain.com etc but will NOT run from otherdomain.com, www.otherdomain.com, otherdomain.net etc.

Example 2: www.mydomain.com

Script will run ONLY from www.mydomain.com. It will not run on the main domain mydomain.com and all other subdomains like myname.mydomain.com as well as other domains like otherdomain.com, www.otherdomain.com, otherdomain.net etc.

You may use * and ? wildcards when specifying a domain name. Wildcards have their usual behavior similar to a file system.

Example 3: *.mydomain.com

The script will run from www.mydomain.com, extra.mydomain.com, myname.mydomain.com etc but will NOT run from mydomain.com, otherdomain.com, otherdomain.net etc.

Example 4: *mydomain.com (please note a change from the previous example)

The script will run from www.mydomain.com, extra.mydomain.com, myname.mydomain.com etc. AND mydomain.com but will NOT run from otherdomain.com, otherdomain.net etc.

Locking to domain names works only for scripts which will run on web servers. As there is no definite domain name when the script runs from shell (command line) you should NOT use locking to domain name for scripts which will run from shell e.g. for cron jobs.

`--domain-encrypt [domain]`

Lock and encrypt scripts to one domain. You may specify only one domain. The encoder will lock the script to run only from the specified domain name. The encoder will use a specified domain name as a part of the key for encryption for providing maximum protection. The Loader will not be able to even decrypt the script that runs in the wrong domain name and will display an error message: "Protected script checksum error". You should not use wildcards for domain name when using this option. This option works ONLY for one strictly specified domain name. You may use this option ONLY ONCE in the command line. This option may not be used with the `--domain` option at the same time.

Locking to domain names works only for scripts which will run on web servers. As there is no definite domain name when the script runs from shell (command line) you should NOT use locking to domain name for scripts which will run from shell e.g. for cron jobs.

Be careful when using this option if you may possibly need to run your protected script from a sub domain. Example: `--domain-encrypt mydomain.com` will allow to run script ONLY from mydomain.com not even from www.mydomain.com or vice versa.

`--ip [x.x.x.x{/y.y.y.y}]`

Lock scripts to IP/mask. The encoder will lock the script to run only from the specified IP address(es). A specified IP address mask will be applied to the real IP address before comparing. So you may use this option to lock the script to a subnet if a correct mask is specified. If a protected script is run from the IP address which is not allowed, the script terminates with the error message: "This script is not licensed to run on this machine". You may use this option more than once to specify multiple IP/mask pairs. This option may not be used with `--ip-encrypt` option at the same time. IP address mask 255.255.255.255 is used by default if not specified.

Locking to IP addresses works only for scripts which will run on web servers. As there is no definite IP address when the script runs from shell (command line) you should NOT use locking to IP address for scripts which will run from shell e.g. for cron jobs.

`--ip-encrypt [x.x.x.x{/y.y.y.y}]`

Lock and encrypt scripts to IP/mask. The encoder will lock the script to run only from the specified IP address. The encoder will use a specified IP address with an applied mask as a part of the key for encryption for providing maximum protection. A SourceGuardian Loader will not be able to even decrypt the script that runs from the wrong IP address and will display an error message: "Protected script checksum error". You may use this option ONLY ONCE in the command line. This option may not be used with `--ip` option at the same time. IP address mask 255.255.255.255 is used by default if not specified.

Locking to IP addresses works only for scripts which will run on web servers. As there is no definite IP address when the script runs from shell (command line) you should NOT use locking to IP address for scripts which will run from shell e.g. for cron jobs.

`--mac [xx:xx:xx:xx:xx:xx]`

Lock the script to LAN hardware (MAC) addresses (The "MAC" word used here has nothing about Apple Macintosh computers. MAC address is a hardware address of the local area networking LAN controller available for all platforms). This address is usually unique for each networking adapter and so it may be easily used to identify a machine. A MAC address is 6 bytes long, with each byte represented in hex and separated with a colon (:). The encoder will lock a script to run only from the machine which has a networking adapter with a specified MAC address. If there is more than one LAN adapter installed then script will check all of them. If an attempt is made to run the script on a machine that is missed a correct adapter, then the script fails with an error message: "This script is not licensed to run on this machine". You may specify multiple MAC addresses, if any one address matches then the script runs.

You may use 'ifconfig' command in Linux or OSX or 'ipconfig /all' in Windows to get a list of installed networking adapters and know MAC addresses.

`--external [filename]`

Scripts encoded with this option will require a license file in order to run. A license file may be deployed with protected scripts or separately from them. This option gives you an opportunity to encode your scripts once and deploy to users providing them with different licenses. Each license may have different locks within it. A license file is not created during encoding. You should use SourceGuardian [license generator](#) (licgen) tool for creating a license file for the script or you may do it in the GUI application. This option can be used only ONCE in the command line. This option may not be used with any other locking options.

If the name of the license file does not include any path then protected scripts will search for the license file in the protected script's folder, its parent folder and so on. So you may have one license file for the entire project located in the top project's folder. A path may be specified for a license file. It may be an absolute file system path or URL like `http://xxx/mylicense.lic` (you may use any name for the license file). If a protected script cannot find the specified license file it will display the following error message: "script requires ... file to run".

Since version 10 of SourceGuardian loaders you may use `SG_LIC_PATH` environment variable to specify where the loader should search for the license file. If the `SG_LIC_PATH` variable is set then the loader will not use any other methods of searching the license file. The environment variable should be available for PHP engine, this may require changes in the web server's configuration.

Example: `--external script.lic`

Example: `--external http://myserver.com/user123.lic`

If you chosen to lock your scripts to an external license file this gives some additional protection to your scripts. Using locking to a license file is the best if you need to deploy one script or the entire project to different users, but need to use different restriction options for each of them.

`--projid [project_id_string]`

Allows you to specify Project ID to identify your project. This option is required if `--external` option is used. When generating a license file using [licgen tool](#) or GUI you need to specify the same Project ID/ Project Key pair. This option is useful when you want to deploy several products that use external license locking so that each license works only with a corresponding Project ID.

`--projkey [project_key_string]`

This option is used in conjunction with Project ID and required if `--external` option is used.

Introduced in version 5.0 a new algorithm uses an idea of two keys. The first key (Project Id) is stored within the encrypted area of protected scripts and used to decrypt an external license file. The second key (Project Key) is stored within the license file and used to decrypt the bytecode from the protected script.

This algorithm protects your product against creating a full working copy from the demo version by some people who may be interested in this. In order to decrypt and run the protected script a valid license file for the full version of your product is required. Otherwise it's impossible to decrypt and run the bytecode. Both Project Id and Project Key values are required if `--external` option is used.

`--conj`

This option makes sense only when encoding multiple files. All scripts encoded with this option will work only with other encoded files of the same project and will NOT work if any of the included files or top files are substituted with an unencoded one or encoded as a part of another project or by another installation of SourceGuardian™ for PHP. This gives you the ultimate protection for your projects when multiple PHP scripts are used together.

Example: If you have a password in a.php and then b.php includes a.php and calls c.php for any action. Enabling this option makes it impossible to substitute c.php with their own code and do 'echo \$password' to know your password. Also enabling this option makes it impossible to create d.php which includes protected a.php and then does 'echo \$password'.

NOTE: Since SourceGuardian 5.0 this option was changed to allow including and executing only scripts from the same project (with the same Project ID value). This lets you develop and encode parts of your project on multiple machines (with multiple SourceGuardian licenses) and keep the "conjunction" option on for maximum protection.

We recommend to always use this feature if your project has any secure data embedded in scripts such as usernames, password, database names etc.

Since SourceGuardian 5.0 the "conjunction" option is always applied to the script during encoding and not to the external script license.

4.2.5 Advanced options

`--asp-tags`

Enables use and recognition of ASP-like `<% %>` tags for indicating the PHP code in addition to standard `<?php ?>` tags.

`--no-short-tags`

Disables use and recognition of short PHP tag `<?` for indicating the PHP code. Both standard `<?php ?>`

and short `<? ?>` tags are recognized by default.

`-p "code"`

Custom header. This options lets you add a custom header at the top of every encoded file. You may put any code to be executed BEFORE the protected scripts code. This code WILL NOT BE ENCODED (although it is still protected with CRC against changes). This may be either HTML text or PHP code. For PHP code - you should enclose your code with `<?php ?>` tags. This option is usually used for including copyrights into protected scripts but also it's used for including [custom error handler](#) functions. Prepend all double quote characters with a back slash if you want to include them into the code (" - > \ ").

Example 1:

`-p "<!-- My protected script. Copyright by \"My Name\" -->"`

Example 2:

`-p "<span class=\"bold\">My protected script. Copyright by My Name</span>"`

Example 3:

`-p "<?php echo \"My protected script. Copyright by My Name\"; ?>"`

You may load the contents of a custom header from the file. Replacing double quotes is not needed in that case.

Example 4:

`-p @my_custom_header.php`

`-j "code"`

Change loader not found code. It is possible to change the default action when an appropriate SourceGuardian loader is not installed and could not be found or used for automatic dynamic loading. The default handler included into protected script starter's code (which prepends each protected script by default) will display an error message "This script is protected by SourceGuardian™ and requires file ... " and then the script stops executing. This option allows you to change the default error action. You may use any HTML text or PHP code and it will be displayed or executed as a replacement to the default SourceGuardian™ loader error. This code WILL NOT BE ENCODED (although it is still protected with CRC against changes). This may be either HTML text or PHP code. For PHP code - you should enclose your code with `<?php ?>` tags. Prepend all double quote characters with a back slash if you want to include them into the code (" -> \ ").

Example 1:

`-j "<a href=\"email:admin@domain.com\">Contact administrator</a>"`

Example 2:

```
-j "<?php header("Location: /myhandler.php"); exit(); ?>"
```

You may load the contents from a file. Replacing double quotes is not needed in that case.

Example 3:

```
-j @my_loader_not_found.php
```

Custom header code and Loader error code may be loaded from a file

Custom header code option -p and Loader error code option -j may load the source from a file. Use @filename as a parameter for -p or -j.

Example: sgencoder -p @prepend.php -j @loadererr.php myscript.php

The code is loaded during encoding and stored as is *non encoded* because that code is executed when no SourceGuardian loader is loaded or when protected script's bytecode is not decoded yet.

-n

Don't integrate default starter code. You may use this option if you don't want to include the default starter code into protected scripts. Scripts encoded using this option will not be able to automatically find and load an appropriate SourceGuardian loader and you have to install the ixed loader manually to run this script. See this [section](#) about the manual ixed installation. If you already have the SourceGuardian loader installed server-wide in php.ini then this option may be useful.

Since PHP 5.2.5 dynamic extensions including SourceGuardian loaders must be installed to PHP's extension_dir folder specified in the php.ini configuration file and an appropriate extension=ixed.XX.YYY directive must be added to the php.ini in order to install the loader.

Note: if you select this option then "Loader not found error code" (-j) option has no effect (as the code is placed inside the default starter code).

-z[0-9]

Specify compression level. Higher compression level gives smaller output scripts which run faster but encoding process will be slower (and vice versa).

--strict-errors

Report E_STRICT compiler errors. E_STRICT "Strict Standards" warnings were introduced in PHP 5. This option instructs the encoder to warn of such messages during encoding. This option is ignored when encoding scripts in PHP 4.x mode. Usually E_STRICT warnings may be ignored but it may be a good idea to let the encoder display such warning messages and review the code. The encoder will stop if --stop-on-error option is also specified.

Note: the encoder can catch only compiler-related E_STRICT warnings. Run-time E_STRICT messages will be displayed when the protected script runs as usual according to the error_displaying option in the php.ini.

`--deprec-errors`

Report `E_DEPRECATED` PHP 5.3+ compiler errors. `E_DEPRECATED` warnings were introduced in PHP 5.3. This option instructs the encoder to warn of such messages during encoding. This option is ignored when encoding scripts for PHP older than 5.3. Usually `E_DEPRECATED` warnings may be ignored but it may be a good idea to let the encoder display such warning messages and review the code. The encoder will stop if `--stop-on-error` option is also specified.

Note: the encoder can catch only compiler-related `E_DEPRECATED` warnings. Run-time `E_DEPRECATED` messages will be displayed when the protected script runs as usual according to the `error_displaying` option in the `php.ini`.

`--stop-on-error`

Stop on compiler errors. This option instructs the encoder to stop encoding at first critical error or `E_STRICT`/`E_DEPRECATED` warning if appropriate options are selected. This may be useful if you have many files in the project and there is a risk of missing errors and leaving some files unencoded because of it.

Note: Even if this option is off error messages will be printed to console during encoding.

`--eval-compatible`

Enables `eval()` compatibility for encoded scripts. Normally encoded scripts cannot be run with the PHP `eval()` function. Additional CRC check restricts it as the protected code is passed as a string and source file is unknown. This improves security for encoded scripts that run in a standard way. However, some PHP template engines or custom code requires loading encoded PHP scripts as a string and then passing it to the `eval()`. In order to enable running protected scripts with `eval()` you may use this option and encode those files in the 'eval compatibility' mode.

`--keep-file-date`

This option instructs the encoder to keep the modification date for encoded files the same as modification date of source files. This may help in deploying only updated files and in some other cases of custom deployment of encoded files. The modification date for encoded files is set to the current date by default if this option is not used.

4.2.6 Custom predefined constants

SourceGuardian lets you define custom named constants during the encoding process, or within an external script license. Constant name/value pairs are stored internally in the encrypted area of the protected script or the license file. They may be used for custom script locking or other actions if you need to store a custom value in protected scripts or a script license file and then retrieve it from your protected PHP code.

Use `--const name=value` option for `sourceguardian` or `licgen` command. Use quotes if your constant name or its value contains any spaces or other special symbols:

```
sourceguardian --const "licensed_for=Robin Hood" myscript.php
licgen --const "licensed_for=Robin Hood" script.lic
```

You may define only one constant with each `--const` option. Add as many `--const` options as you need into the command line.

To get a predefined constant value from the protected PHP code use `sg_get_const()` API function. This function is defined in the SourceGuardian loader.

Syntax: `string sg_get_const( string )`

Returns a predefined SourceGuardian constant value or FALSE if a constant with the specified name is not defined. SourceGuardian constants **names are case sensitive**.

There are 5 predefined constants for all protected scripts:

<code>sg_get_const("encoder")</code>	Returns the name of the encoder - "SourceGuardian"
<code>sg_get_const("version")</code>	Returns version number of the encoder
<code>sg_get_const("encode_date")</code>	Returns UNIX timestamp for the date when the script was encoded
<code>sg_get_const("license_date")</code>	Returns UNIX timestamp for the date when the script license was created. It's may differ from the "encode_date" when an external script license is used
<code>sg_get_const("expire_date")</code>	Returns script expiration date as UNIX timestamp if it's defined in the script license or internally with in the script during encoding

4.2.7 Custom error handling

You may add custom error handling functions which catch script licensing errors. Error handler must be a function which accepts two parameters:

```
sg_error_handler( code , message )
```

You may use any name for this function. Also you may have different functions for different script errors. The first argument will contain an error code. The second one will contain a default error message.

To set a custom error handler, use `--catch` option in sourceguardian command:

```
sourceguardian --catch err=function myscript.php
```

Where "err" is one of predefined constants and "function" is your error handler function name.

Err	Code	Default message
ERR_LICENSE	01,02,03	This script is not licensed to run on this machine...
ERR_EXTLICCR06		A license file which is required to run this protected script is invalid...
C		
ERR_EXPIRED	09	This script has expired...
ERR_EXTLIC	13	This protected script requires ... license file in order to run
ERR_OFFLINE	20	This protected script requires the Internet connection in order to run
ERR_ALL	-	-

"ERR_ALL" is a special value to specify one error handler function for all SourceGuardian error codes.

Custom error handler function should be defined before an error may occur. The best place for it is in the custom header code (see -p option) as it's loaded **before** any license checking is done and so error handlers will be always available if defined there. But you may also define a custom error handler function in another encoded file which is included before the script which may cause a license error. Don't put any passwords etc secret data to your custom header code as this code is stored unencoded within protected scripts, yet it is still protected by CRC against modifications.

Custom error handling with standard PHP error handling mechanism

You may catch some SourceGuardian errors using standard PHP error handling mechanism. This may be useful if you already have an error handler in your code. Below is an example of an error handler to catch SourceGuardian errors.

```
<?php
function myErrorHandler($errno, $errmsg, $filename, $linenum, $vars) {
    if ($errno & E_NOTICE) return;
    if (strstr($errmsg, 'SourceGuardian')) {
        $code = substr($errmsg, strpos($errmsg, '[')+1,2);
        echo "SourceGuardian error $code"; // replace this with what you need for SourceGuardian errors
    } else
        echo $errmsg; // replace this with what you need for other PHP errors
    }
    error_reporting(E_ERROR);
    set_error_handler("myErrorHandler");
?>
```

4.2.8 Excluding files by the file mask

You may exclude some files or directories from encoding when use the command line encoder. Please use --exclude=mask option to specify file(s) and/or dir(s) to exclude from processing. You may specify either a strict name, relative path with a directory name or a mask (with ? and/or * wildcard symbols).

UNIX users: **Always quote masks under UNIX.** Otherwise your shell interpreter will replace the specified file mask with real file and directory names and the result may be unexpected. You should always quote masks that specifies files to encode too (like "*.php" in the example below).

Example: sourceguardian -r --exclude "doc/*" --exclude "config.php" "*.php"

This will encode all *.php files in the current directory and all directories recursively but all files in the "doc" directory and all files (and dirs if any!) named "config.php" will not be encoded.

You may enumerate all the files you want to exclude from encoding using a file list to specify multiple files. A file list is a text file with either full or relative file paths of all the files to exclude, separated by a new line (masks are supported, use * and/or ? wildcard symbols). Specify the @ sign before the filelist name in the command line, e.g. -x @filelistname

4.2.9 Excluding files from encoding but still copying to output directory

We have added an option into the command line encoder to specify which files should be encoded (-f). You may specify what files will be encoded specifying their filenames, filemasks or filelist. All other files which have been added for processing or found by expanding filemasks will be copied to the output

directory "as-is" without encoding. If you don't specify the -f option then all specified files will be encoded by default.

Example 1:

```
>sourceguardian -r -f "*.php" -o "output_dir" ""
```

All (with recursion) *.php files from the current directory will be encoded and copied to output_dir. All other files from the current directory will be copied to output_dir as-is (unencoded).

You may specify multiple filenames or filemasks adding more than one -f option:

Example 2:

```
>sourceguardian -r -f "*.php" -f "includes/*.inc" -f @myphpfiles -o "output_dir" ""
```

If you don't specify the output directory but use -f option then only files specified with -f option will be encoded. All other files will remain unchanged.

4.2.10 Encoding directory content without specifying filemasks

It's possible to use shorter syntax for directory encoding. All specified directories will be recognized and the "*" filemask is added:

```
>sourceguardian -r source_dir
```

instead of the following in old versions of SourceGuardian:

```
>sourceguardian -r "source_dir/*"
```

4.2.11 Output directory for encoded scripts

You can specify an output directory for all encoded scripts when encoding from command line. Source files will be unchanged if you specify the output directory and it differs from your source directory. The default backup option will be off when the output directory is specified. If you want to re-enable it, even when the output directory is specified, then use the -b <backup_extension> option after the output directory option.

Carefully specify the output folder which should never overlap with your source. The command line encoder does not do any checking for that.

The full directory path to source scripts will be recreated under the output directory if the full path to source files was specified. Windows users - drive names ("C:", "D:", etc) will be replaced with just one letter ("C", "D", etc) when recreating the path under the output directory.

Command line option: -o <output_dir>

Example 1: Encode all *.php scripts in the current directory with recursion and put encoded files to /home/myproject/encoded.

```
>sourceguardian -r -o /home/myproject/encoded "*.php"
```

Example 2: Encode all scripts specified in the filelist and put encoded files to /home/myproject/encoded. Additionally backup source scripts in the source directory with .bak extension.

```
>sourceguardian -o /home/myproject/encoded -b bak @filelist
```

4.2.12 Encoding to standard output

SourceGuardian encoder may be used for encoding separate files taking source from standard input and sending encoded contents to standard output. In order to use the command line encoder in this mode, pass the -- (double dash) instead of the input file name.

For example:

```
>sourceguardian -V5 -- < /path/to/source.php > /path/to/encoded.php
```

4.3 Script license generator (full version)

SourceGuardian Script License Generator is a tool for creating script license files. A script license file is required to run protected scripts if they were encoded with the --external option or when a license file name was specified in GUI.

Using locking to a script license is the best way of encoding if you need to distribute one script or your entire project between different users but need to use different restriction options for each of them. What you need to do is encode your scripts with the --external option (or specify a license file name in GUI) and then create a license for each user using the Script License Generator.

Scripts encoded with the --external option will require a license file in order to run. Protected scripts search for the license file in the script's directory and then parent directories. This lets you have one license file located in the top project directory for the entire project.

If protected scripts cannot find the specified license file, an error message is displayed: "This protected script requires ... license file in order to run"

When you bind protected files to a license file the following algorithm is used. This algorithm uses the idea of two keys. The first key (Project Id) is stored within the encrypted area of the protected script and is used to decrypt a license file. The second key (Project Key) is stored in the license file and it is used to decrypt a bytecode stored within the protected script.

The above algorithm effectively protects your php files against license file substitution which otherwise might be used for creating a full working copy from the demo version etc. A valid license file for the full version of your product is required in order to decrypt and run protected script. If the license file is not found or if it's invalid, protected bytecode cannot be decrypted and run.

Project Id and Project Key values are required for binding protected files to the license file. Please specify Project Id (--projid) and Project Key (--projkey) values using options in the command line for the "sourceguardian" command. Project Id and Project Key may be any words, numbers or random strings but for security reasons these two values should not be calculated from each other. Also you must specify the same Project Id/Project Key pair when running the "licgen" command for generating a license file for your protected files. Different projects must use unique Project Id/Project Key values unless you want to share the same license files between projects.

Command line example:

```
sourceguardian --external script.lic --projid "19Gh42Ki" --projkey "Ab65qZ32" myscript.php  
licgen --projid "19Gh42Ki" --projkey "Ab65qZ32" --days 7 script.lic
```

If you have licenses for multiple SourceGuardian installations you may encode scripts on one machine and generate license files on another machine. You must specify the same Project Id and Project Key values for your project and the license generator on different machines.

Error messages

If any protected script runs with an incorrect license file the following error message appears:

"Fatal error: A license file which is required to run this protected script is invalid. Contact the script author to get a license file. Error code [06]"

If the script runs with a license file created with a correct Project Id but incorrect Project Key (this may be a script hacking attempt or accidental modification of the license file or the script) the following error message appears:

"Fatal error: SourceGuardian Loader - Protected script checksum error. The file has been modified. If this script requires a license file in order to run, this error may be caused by an invalid license file. Install the original unmodified file or contact the script author to get the original file or license file. Error code [12] in ..."

Important Security Notice

(!) Safely keep your Project Id and Project Key values.

(!) Remember your Project Id and Project Key. It's impossible to restore the values. They are required for generating licenses for your customers. If you use GUI - Project Id and Project Key are stored within SourceGuardian project files.

(!) When generating Project Id and Project Key manually, please use random values that cannot be guessed or calculated from each other.

Encoding without locking to a license file

You may still encode your scripts and use an embedded license. In this case Project Id and Project Key values are not required. However, Project Id is required for "conjunction" feature ([see above](#)).

4.3.1 Usage

Running the command line license generator included in GUI

If you are using the command line license generator installed with GUI, please please always specify a full path to the SourceGuardian license file (encode.lic) using the -L /path/to/encode.lic command line option. It's not needed if you installed the command line encoder and tools separately and the license file was created in the command line encoder's installation folder. Please refer to the [command line encoder usage section](#) to know standard paths to the encode.lic license file.

License generator command line options

licgen -L/path/to/encode.lic [options] output.lic

--expire <dd/mm/yyyy>	Set script expiration date
--days <nn>	Set script expiration days (from today)
--domain <domain>	Bind script to domain name
--ip <x.x.x.x[/y.y.y.y]>	Bind script to ip/mask
--mac <x:x:x:x:x:x>	Bind script to mac address
--conj	Work only with other encoded files
--projid <value>	Set project id (required, the same as for encoding)
--projkey <value>	Set project key (required, the same as for encoding)
--const name=value	Set custom defined constant
--time-server <server,server,...>	Set time server (for expiration date check)
--compat	Compatibility mode with SourceGuardian 4.x
--text "text"@file	Add plain text into the license file
-l	Display SourceGuardian license information for the tool itself
-w	Wait for key press before exit
-v	Display version number
-h	Display options help

output.lic - this is a name of the license file to generate. It should be the same that you used in --external option during encoding.

Since version 9 of SourceGuardian you may use -- (double dash) instead of the output file name in order to send licgen's output to console instead of a file which may be useful for automating license generation when running licgen on the server side.

Locking options

Most of the options work exactly the same as binding options of SourceGuardian™ 10 for PHP. Please refer to the [Script locking options](#) section for details.

Options unique to the license generator

--text "text"

This option lets you add custom text that will be embedded as-is into the license file. The text is protected with a checksum against modification. You may include any text such as user information, license description etc.

All user {constants} that are defined in [locking options](#) or with [--const option](#) in the command line will be replaced in the text. Also some standard SourceGuardian constants may be used:

{SG_DATE} - current date i.e. date of encoding
 {SG_LICENSEE} - SourceGuardian license owner from the SourceGuardian license file

It works in the same way also for the custom header in protected scripts. [See details](#)

4.4 File information tool (full version)

It's possible to get information about protected scripts or a script license files using SourceGuardian file information tool. This may be useful for supporting your customers, checking licensing issues etc. You may know the date of encoding, expiration date, binding options from the protected script or a script license.

If you use GUI, you may use [File Information](#) option.

Command line encoder users: please use the 'sginfo' tool which is installed along with the encoder. You may pass encoded script name or script license as a parameter to this tool.

Additionally you may need to specify a project key (--projkey), target ip (--tag-ip) and/or target domain (--tag-domain) to let the script information tool decrypt the encoded file and display information. Also you need to specify the project id (--projid) value to decode and display script license information.

It's possible to display script information only for files created with the same installation of SourceGuardian.

4.4.1 Usage

Getting encoded script's information

sginfo [options] file.php

--tag-ip [x.x.x.x{/y.y.y.y}]	Target IP for decryption
--tag-domain [domain]	Target Domain for decryption
--projkey [value]	Script Project Key for decryption

Getting license file information

sginfo [options] file.lic

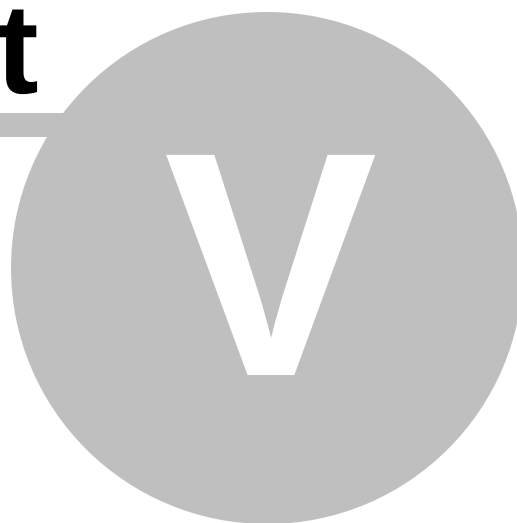
--projid [value]	License Project ID for decryption
------------------	-----------------------------------

General options

-v	Display version number
-l	Display SourceGuardian license information for the tool itself
-h	Display options help

SourceGuardian 10 for Mac OS X

Part



5 Encoding of HTML templates and other non-PHP files

You may encode HTML templates, custom configuration files or other non-PHP files using the SourceGuardian encoder. HTML templates or other non-PHP files may be encoded by the encoder and then read and decrypted from the protected PHP code using SourceGuardian API function `sg_load_file()`. Please refer to the [section below](#) to know how you can use encoded templates from your encoded PHP files. Files encoded in this mode cannot be automatically run by the PHP engine.

Encoded templates look like this:

```
SourceGuardianAAwAAAAFCgAAAAZ0jwEA/9QAMUp+g+GpvG3vbvYj4Is=
```

Within this document we will refer to HTML templates or other non-PHP files simply as "templates" for short. There is no difference for the encoder between HTML templates, other templates or any other non-PHP files.

Template files encoded as a part of a project may be used only from protected scripts which were encoded as a part of the same project. It's impossible to use protected templates from unencoded scripts or from scripts encoded as a different SourceGuardian project.

Internal `project_id` and `project_key` values are used for identifying the project and used as a key for encoding templates. So please make sure to specify the `project_id` (`--projid` option) for the command line encoder as well as the `project_key` (`--projkey` option) for the project and external script license when generating a license with `licgen` tool. Simply always specify the `project_id` for all your projects (unique for each) and additionally the `project_key` when locking to an external license is used.

SourceGuardian GUI generates `project_id` and `project_key` automatically for a new project. `Project_id` and `project_key` are saved within your SourceGuardian project file. Please always use the same project for adding/changing encoded templates otherwise old templates cannot be used with newly encoded scripts or vice versa because of new `project_id` and `project_key`. Save your `project_id` and `project_key` values for future use. The `project_key` value is also required for correct license generation if you use locking to a license file.

5.1 GUI

Encoding templates is simple in SourceGuardian GUI. In the [project window](#) choose "Template" type for files or folders that you would like to encode as templates. These files will be encoded as templates when you click Encode.

5.2 Command line interface

Use the `-t` option to specify files, file masks or file list for template files.

Example 1:

```
>sgencoder -r -t"myproject/templates/*.tpl" "myproject/"
```

You may specify multiple `-t` options if you need. All other files which are not specified as templates will be encoded as PHP scripts.

If you use `-f` option (see below) to specify files to encode then files specified by `-f` options will be encoded

as PHP scripts, files specified by `-t` options will be encoded as templates and all other files will not be encoded and will be copied to the output directory as-is. Output directory may be specified by the `-o` option.

You may use file lists for specifying template files and use file masks as well as normal file names in the list.

Example 2:

if 'mytemplates' file contains:

```
*.tpl  
*.html  
*.htm  
templates/mytemplate.txt
```

```
>sgencoder -r -t @mytemplates -o output_dir source_dir
```

This command will encode files specified in 'mytemplates' (i.e. templates/mytemplate.txt, *.tpl, *.html and *.htm files in source_dir) as templates, it will encode all other files in source_dir as PHP scripts.

Example 3:

if additionally 'myphpfiles' file contains:

```
*.php  
*.inc
```

```
>sgencoder -r -t @mytemplates -f @myphpfiles -o output_dir source_dir
```

This command will encode *.tpl, *.html and *.htm files in source_dir as templates, *.php and *.inc files as PHP scripts and will leave all other files from source_dir unencoded but copied to the output_dir. See details below about `-f` option.

5.3 Using protected templates

An encoded template may be loaded from the protected script using the `sg_load_file($filename)` SourceGuardian API function. It returns decoded file contents as a string or generates an error.

```
$template_data = sg_load_file($filename);
```

`sg_load_file()` may generate the following errors:

SourceGuardian Loader - Encoded template file ... is not found. Contact the script author about this problem. Error code [21]
(when the loader could not find a specified template file)

SourceGuardian Loader - Incompatible loader version when loading encoded template file ... Please download and install the latest loaders. Error code [22]
(you are trying to load a template encoded with a newer version of the encoder but have an older loader installed)

SourceGuardian Loader - Decryption error for encoded template file ... Install the original unmodified file or contact the script author to get the original file. Error code [23]
(an error has been detected at the decoding stage, possibly because the template that you are trying to load was encoded for another SourceGuardian project - different project_id or project_key)

SourceGuardian Loader - Error loading encoded template file ... Check file permissions or contact the

script author about this problem. Error code [24]
(system error when loading a template file - insufficient memory, read error etc)

All errors are E_USER_ERROR and may be caught by a custom error handler.

5.4 Using protected templates with the Smarty template engine

We have created an updated version of the Smarty template engine which can read encoded templates. This version is available from our site <http://sourceguardian.com/scripts/Smarty-2.6.14-SG.tar.gz>. The current version, as of writing this document, is 3.0 but it should be easy to update other versions too. Please read details below about the changes we have done:

To enable loading of encoded *.tpl files the following simple changes are required:

Smarty.class.php

```
function _read_file($filename)
{
    //SourceGuardian patch
    if ( function_exists("sg_load_file") ) {
        if ( file_exists($filename) ) {
            return sg_load_file($filename);
        } else {
            return false;
        }
    }

    if ( file_exists($filename) && ($fd = @fd($filename, 'rb')) ) {
        $contents = "";
        while (!feof($fd)) {
            $contents .= fread($fd, 8192);
        }
        fclose($fd);
        return $contents;
    } else {
        return false;
    }
}
```

To enable additional protection of recompiled template files the following additional changes are required:

In Smarty.class.php function fetch() and function _smarty_include()

replace:

```
include($_smarty_compile_path);
```

with:

```
//SourceGuardian patch
sg_eval(sg_load_file($_smarty_compile_path));
```

In `internals/cordite_file.php` function `smarty_core_write_file()`

replace:

```
if (!$fad = @fopen($_tmp_file, 'wb')) {
    $_tmp_file = $_dirname . DIRECTORY_SEPARATOR . uniqid('wrt');
    if (!$fd = @fopen($_tmp_file, 'wb')) {
        $smarty->trigger_error("problem writing temporary file '$_tmp_file'");
        return false;
    }
}

fwrite($fd, $params['contents']);
fclose($fd);
```

with:

```
//SourceGuardian patch
if ( function_exists("sg_encode_file") ) {
    sg_encode_file($_tmp_file, $params['contents']);
} else {
    if (!$fd = @fopen($_tmp_file, 'wb')) {
        $_tmp_file = $_dirname . DIRECTORY_SEPARATOR . uniqid('wrt');
        if (!$fd = @fopen($_tmp_file, 'wb')) {
            $smarty->trigger_error("problem writing temporary file '$_tmp_file'");
            return false;
        }
    }

    fwrite($fd, $params['contents']);
    fclose($fd);
}
```

After all the above changes are done the Smarty engine can work with normal unencoded templates when runs from unprotected scripts and encoded templates when runs from SourceGuardian encoded scripts. It is not required to encode the Smarty engine itself - this is optional and does not affect the security of your protected scripts or templates.

5.5 Creating custom encoded files from protected scripts

If your script generates files online and you need to secure them, it's possible with SourceGuardian. You may use `sg_encode_file($filename, $data)` SourceGuardian API function for encoding a file from protected code. This file will be encrypted in the same way as the SourceGuardian encoder encodes template files.

```
sg_encode_file($filename, $data);
```

Security notice. A built-in SourceGuardian API encoder (`sg_encode_file()` API function) is suited only for encoding templates, configuration, data files and other non-PHP files. It does not perform compilation into bytecode and should not be used for securing source PHP scripts. Always use the SourceGuardian encoder for protecting PHP scripts as only bytecode compilation with encryption and compression can

give maximum security for your PHP source scripts.

`sg_encode_file()` may generate the following error:

SourceGuardian Loader - Error writing file ... Check file permissions or contact the script author about this problem. Error code [25]

(The loader failed to create an output file because of permissions, disk space etc problems)

This error is `E_USER_ERROR` and may be caught by a custom error handler.

Files encoded using the `sg_encode_file()` SourceGuardian API function may be read by the `sg_load_file()` SourceGuardian API function described above.

Files get encrypted using the current protected script's project identifier (`project_id`) and key (`project_key`) and so may be read only by the protected script encoded in the same SourceGuardian project.

If your protected script was encoded using advanced `ip_encrypt` or `domain_encrypt` options then the protected template file written by `sg_encode_file()` will be additionally encrypted using the current IP address (or domain name) as a key. These protected templates can be decrypted only on the machine with same IP (or domain name).

Since SourceGuardian version 8.1 we have changed the way how script keys based on Project ID or Project Key (if an external license file is used) are used for encoding or decoding files with `sg_load_file()` and `sg_encode_file()` loader API functions. Now we keep track of what protected PHP script uses which key and the loader will use an appropriate key for encryption/decryption user files and strings (see new `sg_encode_string()` and `sg_decode_string()` API functions). Older versions of the loader would use the key of the last loaded protected script. These changes allow to use the encoding functions correctly in a situation when one protected script is inherited from another protected script and the encoding/decoding functions are called.

5.6 Using encoding SourceGuardian API from unprotected script

SourceGuardian API functions are part of the SourceGuardian loader and they are only available when the SourceGuardian loader is loaded into PHP. This may be done automatically by the run-time loader of the SourceGuardian protected script, or when the SourceGuardian loader is installed server-wide in `php.ini` and loaded when PHP starts.

`sg_load_file()` SourceGuardian API function returns the file's data as-is without decryption when:

- it runs from unprotected script with loader installed server-wide (this is useful for debugging purposes, see below)
- it loads the template or data file which was not encoded by SourceGuardian

`sg_encode_file()` SourceGuardian API function writes the file's data as-is without encryption:

- when runs from an unprotected script with the loader installed server-wide

5.7 Debugging of scripts which work with encoded templates

Usually SourceGuardian API functions are not available until SourceGuardian is loaded by the protected script. The exception to this is when the SourceGuardian loader is installed server-wide in `php.ini`. It may be not obvious how to debug scripts using the SourceGuardian API because of that. For convenience

and easy debugging we suggest two possible ways for debugging scripts which use the SourceGuardian encoding API:

1) Install an appropriate SourceGuardian loader server-wide in `php.ini` as a PHP extension. Usually it's possible to do on a development machine as normally PHP installation is over developer's control there. If the loader is installed server-wide, SourceGuardian API functions will be always available. When called from the unencoded source scripts `sg_encode_file()` and `sg_load_file()` functions are both work with unprotected data for reading and writing and so it's easy to debug and check content of the output file or loaded template file etc. When project debugging is completed and project is encoded, SourceGuardian API functions will start working in normal (protected) mode for reading and writing encoded templates/non-PHP files data.

2) Use our SourceGuardian API stub script `sgapistub.php` (<http://sourceguardian.com/scripts/sgapistub.php>) This is very simple PHP script which simulates `sg_load_file()` and `sg_encode_file()` functions without doing any encoding or decoding. When run from the protected script and SourceGuardian API functions are available this script does nothing and lets real API functions work. To use this stub script you need to include it from your script that uses SourceGuardian encoding API. When running from unprotected script, functions defined in this stub script will read and write templates or data files as-is which lets debug the script and check content of the output file or loaded template file. When project debugging is completed and project is to be encoded with SourceGuardian you need to encode the stub script with all other PHP scripts in your project. When run from the protected script the stub file itself will do nothing as real SourceGuardian API functions will be used. This is done for convenience and you don't need to search your scripts and remove or comment the include directive which includes the stub script. Please read comments in the beginning of `sgapistub.php` script before using.

Please feel free to choose the better way for you for debugging your protected scripts. The second method does not require to have access to `php.ini` even in development environment.

SourceGuardian 10 for Mac OS X

Part



VI

6 Common mistakes

This section includes common mistakes that people may make, either in encoding and protecting their files, or in uploading or running these files on the web server. They are not in any particular order, but we would suggest that you look at this section before you contact SourceGuardian regarding any support matter.

6.1 Encoded scripts modification

Encoded scripts are protected against modification. Please **DO NOT MODIFY** any single byte in the encoded scripts or you will get an error when running them.

6.2 Extension directory (php.ini setting)

If you want the ixed loader to be loaded dynamically using dl() function you have to make sure that *extension_dir* setting in your php.ini is valid. It must point to a directory that does exist on the server. If it doesn't exist then PHP cannot load any extension at all (including the ixed loader).

For windows users only: *extension_dir* option in php.ini should point to the directory located on the same drive with your document root and scripts directory.

6.3 Getting "This protected script can run only in conjunction..." error

Getting "SourceGuardian Loader - This protected script can run only in conjunction with other encoded files of the same project" while using a correct license file and no other non-encoded files are prepended or included from the protected one?

Check that xdebug or another PHP debugging extension is off in the php.ini. Such extensions modify the bytecode on-the-fly and this makes the SourceGuardian loader think that the protected script has been modified or non-SourceGuardian encoded code is running.

SourceGuardian 10 for Mac OS X

Part



VII

7 Advanced users

7.1 PHP shell scripts encoding

SourceGuardian supports encoding of PHP scripts which are UNIX shell scripts. SourceGuardian keeps the first line unchanged for the script if it begins with a `#!/usr/bin/php` UNIX shell script prefix (ex. `#!/usr/bin/php`) This lets encoded scripts run from the shell. The first line of the script will not be encoded but the whole script including this line will be still protected with a checksum and so remains protected from unauthorized modifications. (This also means that the path that may be specified in the first line cannot be changed after the file has been encoded).

7.2 SourceGuardian loader API

SourceGuardian loader defines some functions which are **available from protected scripts**. These functions are available only when the loader is loaded into PHP engine and protected script is running.

array sg_get_mac_addresses()

This function returns an array of hardware addresses (MAC-addresses) of all network interfaces installed on the machine where the script is running. It may be useful for creating custom locking schemas etc. Each MAC address is returned as formatted string that includes 6-byte hardware address in the following format: `XX:XX:XX:XX:XX:XX` Up to 32 network hardware addresses may be returned.

string sg_get_const(\$name)

Returns the value of a custom constant that was during encoding or a value of the predefined constant. See [Custom predefined constants](#) section for details.

string sg_load_file(\$filename)

Loads and decrypts the encoded template and returns contents as a string. See [Using protected templates](#) section for details.

sg_encode_file(\$filename, \$data)

Encodes template or another file from the protected code. See [Creating custom encoded files](#) section for details.

string sg_encode_string(\$data)

Encrypts a string. An internal key based on Project ID or Project Key (if the script is bound to an external license file) is used for encryption. An encoded string is converted to base64 and can be easily saved to a text file. This function and `sg_decode_string()` function are useful for storing critical user settings in encrypted format.

Note: if this function is called from unencoded code it will return unmodified string. This lets you debug your scripts but the actual encoding will work only when the function is called from the protected code.

string sg_decode_string(\$encrypted)

Decrypts a string previously encrypted with `sg_encode_string()`. An internal key based on Project ID or Project Key (if the script is bound to an external license file) is used for decryption. This function and `sg_encode_string()` functions are useful for storing critical user settings in encrypted format.

Note: if this function is called from unencoded code it will return unmodified string. This lets you debug your scripts but the actual decoding will work only when the function is called from the protected code.

7.3 Marking a file to be skipped during encoding

It's possible to mark a file to be skipped by the encoder. Add the following string anywhere in the code, use comments. Skipped files will be copied as-is to the target folder if it's specified.

SourceGuardian:DO_NOT_ENCODE

E.g. `/* SourceGuardian:DO_NOT_ENCODE */`

Note: Comparison is case sensitive for Windows and Solaris if you will ever move to any of these platforms for encoding. Do not change or mix the case for better code compatibility.

SourceGuardian 10 for Mac OS X

Part



8 License

8.1 SourceGuardian License

PLEASE READ THIS CAREFULLY BEFORE USING MATERIALS

A IMPORTANT PROVISIONS

YOUR ATTENTION IS DRAWN PARTICULARLY TO THE PROVISIONS OF CLAUSE 10.2 OF THE FOLLOWING LICENCE AGREEMENT.

B PROPERTY OF SOURCEGUARDIAN

YOU MAY OBTAIN A COPY OF THIS SOFTWARE PRODUCT EITHER BY DOWNLOADING IT REMOTELY FROM OUR SERVER OR BY COPYING IT FROM AN AUTHORISED DISKETTE, CD-ROM OR OTHER MEDIA ('HARD MEDIA'). THE COPYRIGHT, DATABASE RIGHTS AND ANY OTHER INTELLECTUAL PROPERTY RIGHTS IN THE PROGRAMS AND DATA WHICH CONSTITUTE THIS SOURCEGUARDIAN (VERSION 10) SOFTWARE PRODUCT (THE 'MATERIALS'), TOGETHER WITH THE HARD MEDIA ON WHICH THEY WERE SUPPLIED TO YOU, ARE AND REMAIN THE PROPERTY OF SOURCEGUARDIAN LIMITED ('SOURCEGUARDIAN'). YOU ARE LICENSED TO USE THEM ONLY IF YOU ACCEPT ALL THE TERMS AND CONDITIONS SET OUT BELOW.

C LICENCE ACCEPTANCE PROCEDURE

BY CLICKING ON THE ACCEPTANCE BUTTON WHICH FOLLOWS THIS LICENCE AGREEMENT (MARKED 'I ACCEPT'), YOU INDICATE ACCEPTANCE OF THIS LICENCE AGREEMENT AND THE LIMITED WARRANTY AND LIMITATION OF LIABILITY SET OUT IN THIS LICENCE AGREEMENT. SUCH ACCEPTANCE IS EITHER ON YOUR OWN BEHALF OR ON BEHALF OF ANY CORPORATE ENTITY WHICH EMPLOYS YOU OR WHICH YOU REPRESENT ('CORPORATE LICENSEE'). IN THIS LICENCE AGREEMENT, 'YOU' INCLUDES BOTH THE READER AND ANY CORPORATE LICENSEE.

D LICENCE REJECTION PROCEDURE

YOU SHOULD THEREFORE READ THIS LICENCE AGREEMENT CAREFULLY BEFORE CLICKING ON THE ACCEPTANCE BUTTON. IF YOU DO NOT ACCEPT THESE TERMS AND CONDITIONS, YOU SHOULD CLICK ON THE 'REJECT' BUTTON, DELETE THE MATERIALS FROM YOUR COMPUTER AND PROMPTLY (AND IN ANY EVENT, WITHIN 14 DAYS OF RECEIPT) RETURN TO SOURCEGUARDIAN OR A LICENSED RESELLER (A) THE HARD MEDIA; (B) ANY OTHER ITEMS PROVIDED THAT ARE PART OF THIS PRODUCT; AND (C) YOUR DATED PROOF OF PURCHASE. ANY MONEY YOU PAID TO SOURCEGUARDIAN OR AN SOURCEGUARDIAN RESELLER FOR THE MATERIALS WILL BE REFUNDED LESS ANY CREDIT CARD TRANSACTION FEE INCURRED BY SOURCEGUARDIAN.

E OTHER AGREEMENTS

IF YOUR USE OF THESE PROGRAMS AND DATA IS PURSUANT TO AN EXECUTED LICENCE AGREEMENT, SUCH AGREEMENT SHALL APPLY INSTEAD OF THE FOLLOWING TERMS AND CONDITIONS.

LICENCE AGREEMENT AND LIMITED WARRANTY

1. Ownership of materials and copies

The Materials and related documentation are copyrighted works of authorship, and are also protected under applicable database laws. SourceGuardian retains ownership of the Materials and all subsequent copies of the Materials, regardless of the form in which the copies may exist. This licence is not a sale of the original Materials or any copies.

2. Licence

2.1. Evaluation Licence

If you have received an evaluation version of the Materials, SourceGuardian hereby grants to you, strictly for your own internal business purposes (and subject to the other terms and conditions of this Licence Agreement), a limited, non-exclusive licence for a single user to:

2.1.1. install the Materials for use on a single computer owned, leased and/or controlled by you for an evaluation period of 14 days;

2.1.2. make a single copy of the Materials for back-up, archival or other security purposes.

2.2. Full Licence

Provided that you have paid the applicable licence fee (and subject to the other terms and conditions of this Licence Agreement), SourceGuardian hereby grants to you, strictly for your own internal business purposes, a limited, non-exclusive licence for a single user to:

2.2.1. install the Materials for use on a single computer owned, leased and/or controlled by you;

2.2.2. make a single copy of the Materials for back-up, archival or other security purposes.

2.2.3. use SourceGuardian for development, testing, training and demonstration purposes and for the purpose of providing services to end users.

2.3. Subject to the remaining provisions of this Licence SourceGuardian grants to the Licensee a world-wide, royalty free, non-exclusive, licence to permit the Licensee to do the following things in relation to the Loader (being defined as the software program made available on the SourceGuardian website at www.sourceguardian.com in object code form that facilitates the conversion of scripts encoded with SourceGuardian to readable form). The following permissions shall be deemed to apply to and cover any use of the Loaders prior to the effective date of this Licence Agreement.

2.3.1. Distribute free of charge and make copies of the Loader for non-revenue generating activities including but not limited to evaluation, development, demonstration, training purposes, test, verification as well as for end user support. All such copies shall be subject to the provisions of this Licence Agreement;

2.3.2. Merge, incorporate, install and integrate the Loader with any third party or Licensee software;

2.3.3. Use, distribute and market the Loader to end users provided always that end users are either (i) directed to the SourceGuardian website and agree to the terms of SourceGuardian's free Loader Licence or (ii) are supplied with a copy of SourceGuardian's free Loader Licence when the encoded files are supplied.

3. Limited Support

SourceGuardian shall make available to you (at such times and to such extent as SourceGuardian may, in its sole discretion, deem reasonable) limited email support services for a period of 6 months from the date of your first installation of the Materials. Without prejudice to the foregoing provisions of this clause 3, such support services are, in any event, limited to your making a maximum of 20 requests for assistance during the support period.

4. Licence restrictions

You may not use, copy, modify or transfer the Materials (including any related documentation) or any copy, in whole or in part, including any print-out of all or part of any database, except as expressly provided for in this licence. If you transfer possession of any copy of the Materials to another party or use the Materials on a different computer from that on which the Materials were originally installed except as provided herein or without obtaining SourceGuardian's prior written consent, your licence is automatically terminated. You may not translate, reverse engineer, decompile, disassemble, modify or create derivative works based on the Materials, except as expressly permitted by the laws of England and Wales. You may not vary, delete or obscure any notices of proprietary rights or any product identification or restrictions on or in the Materials..

5. No transfer

The Materials are licensed only to you. You may not rent, lease, sub-license, sell, assign, pledge, transfer or otherwise dispose of the Materials, on a temporary or permanent basis, nor use the same for remote hosting, ASP services, to act as a bureau or for time-sharing use without the prior written consent of SourceGuardian.

6. Undertakings

You undertake to:

- 6.1. ensure that, prior to use of the Materials by your employees or agents, all such parties are notified of this licence and the terms of this Licence Agreement;
- 6.2. reproduce and include our copyright notice (or such other party's copyright notice as specified on the Materials) on all and any copies of the Materials, including any partial copies of the Materials;
- 6.3. hold all drawings, specifications, data (including object and source codes), software listings and all other information relating to the Materials confidential and not at any time, during this licence or after its expiry, disclose the same, whether directly or indirectly, to any third party without SourceGuardian's consent.

7. Limited warranty

- 7.1. Subject to the limitations and exclusions of liability below, SourceGuardian warrants that (a) the Hard Media on which the Materials are furnished will be free from material defects under normal use; and that (b) the copy of the program will materially conform to the documentation which accompanies the program. The Warranty Period is 90 days from the date of delivery to you.
- 7.2. SourceGuardian will also indemnify you for personal injury or death solely and directly caused by any defect in its products or the negligence of its employees.

7.3. SourceGuardian shall not be liable under the said warranty above if the Materials fail to operate in accordance with the said warranty as a result of any modification, variation or addition to the Materials not performed by the SourceGuardian or caused by any abuse, corruption or incorrect use or installation of the Materials, including use of the Materials with equipment or other software which is incompatible.

8. No other warranties

8.1. The foregoing warranty is made in lieu of any other warranties, representations or guarantees of any kind, either expressed or implied, including, but not limited to, any implied warranties of quality, merchantability, fitness for a particular purpose or ability to achieve a particular result. You assume the entire risk as to the quality and performance of the Materials. Should the Materials prove defective, you (and not the SourceGuardian nor any licensed reseller) assume the entire cost of all necessary servicing, repair or correction.

8.2. SourceGuardian does not warrant that the Materials will meet your requirements or that its operation will be uninterrupted or error free.

9. Limitation of liability

SourceGuardian's entire liability and your exclusive remedy shall be:

9.1. the replacement of any Hard Media not meeting SourceGuardian's 'Limited Warranty' and which is returned to SourceGuardian together with dated proof of purchase; or

9.2. if, during the Warranty Period, SourceGuardian is unable to deliver replacement Hard Media which is free of material defects, you may terminate this Licence Agreement by returning the Materials to SourceGuardian and any money you paid to SourceGuardian for the Materials will be refunded less any credit card transaction fee incurred by SourceGuardian.

10. Exclusion of liability

10.1. Except in respect of personal injury or death caused directly by the negligence of SourceGuardian, in no event will SourceGuardian be liable to you or any third party for any damages, including any lost profits, lost savings, loss of data or any indirect, special, incidental or consequential damages arising out of the use of or inability to use such Materials, even if SourceGuardian has been advised of the possibility of such damages. Nothing in this Licence Agreement limits liability for fraudulent misrepresentation.

10.2. Without prejudice to any other provisions of this Licence Agreement you hereby expressly acknowledge that encryption software is not infallible and that third parties may develop and employ methods to circumvent the Materials and you agree that SourceGuardian shall have no liability to you or any third party in such circumstances.

11. Your statutory rights

This licence gives you specific legal rights and you may also have other rights that vary from country to country. Some jurisdictions do not allow the exclusion of implied warranties, or certain kinds of limitations or exclusions of liability, so the above limitations and exclusions may not apply to you. Other jurisdictions allow limitations and exclusions subject to certain conditions. In such a case the above limitations and exclusions shall apply to the fullest extent permitted by the laws of such applicable jurisdictions. If any part of the above limitations or exclusions is held to be void or unenforceable, such part shall be deemed to be deleted from this Licence Agreement and the remainder of the limitation or exclusion shall continue in full force and effect. Any rights that you may have as a consumer (ie a purchaser for private as opposed to business, academic or government use) are not affected.

12. Term

The licence is effective until terminated. You may terminate it at any time by destroying the Materials together with all copies in any form. It will also terminate upon conditions set out elsewhere in this Licence Agreement or if you fail to comply with any term or condition of this Licence Agreement or if you voluntarily return the Materials to us. You agree upon such termination to destroy the Materials together with all copies in any form.

13. Export

You will comply with all applicable laws, rules, and regulations governing export of goods and information, including the laws of the countries in which the Materials were created. In particular, you will not export or re-export, directly or indirectly, separately or as a part of a system, the Materials or other information relating thereto to any country for which an export licence or other approval is required, without first obtaining such licence or other approval.

14. General

14.1. You agree that SourceGuardian shall have the right, after supplying undertakings as to confidentiality, to audit any computer system on which the Materials are installed in order to verify compliance with this licence Agreement.

14.2. This Licence Agreement constitutes the complete and exclusive statement of the Agreement between SourceGuardian and you with respect to the subject matter of this Licence Agreement and

supersedes all proposals, representations, understandings and prior agreements, whether oral or written, and all other communications between us relating to that subject matter.

14.3. Any clause in this Licence Agreement that is found to be invalid or unenforceable shall be deemed deleted and the remainder of this Licence Agreement shall not be affected by that deletion.

14.4. Failure or neglect by either party to exercise any of its rights or remedies under this Licence Agreement will not be construed as a waiver of that party's rights nor in any way affect the validity of the whole or part of this Licence Agreement nor prejudice that party's right to take subsequent action.

14.5. This Licence Agreement is personal to you and you may not assign, transfer, sub-contract or otherwise part with this Licence Agreement or any right or obligation under it without the SourceGuardian's prior written consent.

- 14.6. This Licence Agreement and any claim or matter arising under or in connection with this Licence Agreement and the legal relationships established by this Licence Agreement shall be governed by and construed in all respects in accordance with the law of England and Wales, and the parties agree to submit to the non-exclusive jurisdiction of the English courts.

Should you have any questions concerning this Licence Agreement you may contact SourceGuardian Limited at A6 Kinigfisher House, Kingsway, Team Valley Trading Estate, Gateshead, Tyne & Wear. NE11 0JQ United Kingdom. Tel: 0845 155 2455. Email: Support@SourceGuardian.com.

8.2 SourceGuardian Loaders License

This Licence applies to the use of the Loaders for SourceGuardian by End Users and is granted by SourceGuardian Ltd (registered number 05663267) which is referred to in this Licence as "SourceGuardian". The licence terms for the use of SourceGuardian software are set out at <http://www.sourceguardian.com/terms.html>. If you are a licensee of SourceGuardian your use of the Loaders is governed by that licence.

If you are an End User of SourceGuardian then, by downloading the Loader, you are accepting the terms of this Licence on behalf of yourself and any company, unincorporated association or partnership for which you work. This Licence comes into effect on the date that you download the Loader. If, having read the Licence, you do not agree to be bound to any of its terms you should not download the Loader and any enquiries on the content of this Licence should be directed to SourceGuardian Ltd at A6 Kinigfisher House, Kingsway, Team Valley Trading Estate, Gateshead, Tyne & Wear. NE11 0JQ United Kingdom. Tel: 0845 155 2455. Email: Support@SourceGuardian.com.

1. Definitions

Defect

Any material error that prevents the Loader from uploading or downloading scripts to and from SourceGuardian (unless used in conjunction with third party software that is not on the list of maintained software on SourceGuardian's website)

End User(s)

Users of the Loader in commercial operation

Loader

The software program that facilitates the conversion of scripts encoded with SourceGuardian to readable form

Maintainence

Issuing updates to SourceGuardian to ensure continuing compatibility with third party software programs with which SourceGuardian is designed to inter-operate.

SourceGuardian

The software encryption product of that name used to encrypt scripts from human- readable form into a form capable of being read only with the Loader, available at www.SourceGuardian.com

Support

Assisting the End User with enquiries relating to Defects

2. License for End Users

Subject to the remaining provisions of this Licence SourceGuardian grants to the End User a world-wide, royalty free, non-exclusive, licence to permit the End User to use the Loader in its commercial operations for the purposes of:

2.1 Reading scripts encrypted with SourceGuardian;

2.2 Bundling the End User's own software applications together with the Loader;

2.3 Linking the End User's software applications to the Loaders on SourceGuardian's website (to ensure that the applications remain current with the latest updated version of the Loader)

This License shall be deemed to apply to and cover any use of the Loaders prior to the effective date of this Licence.

3. Restrictions and Exceptions

3.1 The rights granted to the End User under this Licence do not operate to assign or transfer the ownership of any intellectual property rights in the Loader to the End User.

3.2 The Loader is intended for commercial use only and not for use by consumers. By accepting this Licence the End User confirms that he or she is acting in the course of business. The End User will not remove or obscure any copyright or ownership notices or warning legends from the Loader nor will the End User attempt to reverse engineer, decompile or otherwise interfere with the Loader except to the extent expressly permitted by law or under this Licence. SourceGuardian may terminate this Licence immediately if it discovers that the End User is in breach of its obligations in this Licence and in such a case the End User will immediately delete the Loader from its computer systems and will on request by SourceGuardian provide and execute such written assurances as SourceGuardian may require confirming such deletion.

3.3 The Loader is licensed free of charge and accordingly is provided on an "as is" basis. The End User agrees and acknowledges that SourceGuardian has no liability to the End User, whether in contract, tort (including negligence) or otherwise arising from any Defects in the Loader and all warranties implied by the laws of any jurisdiction in which the End User uses the Loader are expressly excluded to the fullest extent permitted by the laws of such jurisdiction.

3.4 In no event will SourceGuardian be liable to the End User for any consequential loss or any financial loss.

3.5 SourceGuardian's only obligation to the End User is to use reasonable commercial efforts to (i) correct Defects within a reasonable time (ii) ensure the the Loader is not corrupted and is free of any computer virus, trojans, worms or logic bombs and (iii) to issue Maintenance updates from time to time. SourceGuardian may terminate or assign its obligations under this paragraph 3.5 by publishing a notice to this effect on the SourceGuardian website at www.SourceGuardian.com. In such circumstances the provisions of paragraph 3.2 will also terminate. Nothing in this Licence applies so as to exclude or limit any liability that SourceGuardian may have to the End User based on fraudulent misrepresentation or personal injury resulting from SourceGuardian's negligence.

4. General

4.1 This Agreement contains the entire agreement between the parties on the subject matter of this Agreement and supersedes all representations, undertakings and agreements previously made between the parties with respect to the subject matter of this Agreement.

4.2 If any provision (or part of a provision) of this Licence is found by any court or administrative body of

competent jurisdiction to be invalid, unenforceable or illegal, the other provisions will remain in force. If any invalid, unenforceable or illegal provision would be valid, enforceable or legal if some part of it were deleted, the provision will apply with whatever modification is necessary to give effect to the commercial intention of the parties.

4.3 This Licence constitutes the whole agreement between the parties and supersedes any previous arrangement, understanding or agreement between them relating to its subject matter.

4.4 Each party acknowledges and agrees that in entering into this Licence it does not rely on any undertaking, promise, assurance, statement, representation, warranty or understanding (whether in writing or not) of any person (whether party to this Licence or not) relating to the subject matter of this Licence, other than as expressly set out in this Licence.

4.5 This Licence and any disputes or claims arising out of or in connection with it or its subject matter or formation (including non-contractual disputes or claims) are governed by, and should be construed in accordance with, the law of England and Wales.

4.6 The parties irrevocably agree that the courts of England have exclusive jurisdiction to settle any dispute or claim that arises out of or in connection with the Contract or its subject matter or formation (including non-contractual disputes or claims).

8.3 Other Licenses for SourceGuardian for OSX

This section includes licenses for other products used for creating SourceGuardian™ for Mac OS X.

8.3.1 RegexKit Framework

Copyright © 2007-2008, John Engelhart

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the Zang Industries nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

8.3.2 Sparkle Framework

Copyright © 2006 Andy Matuschak

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

SourceGuardian 10 for Mac OS X

Part



IX

9 Changelog

9.1 Version 10 / June 2014

Version 10 introduces improved code protection methods as well as encoding for PHP 5.6 and some new options. As usual this update is partly based on comments and suggestions of our users. We were glad to receive comments and suggestions and want to thank you very much for sharing your ideas! We are looking forward to hearing about other suggestions for improving SourceGuardian. Here is a list of recent version 10 changes.

NEW FEATURES

- Improved code protection methods
- Full support of PHP 5.6 encoding including the latest language options: constant scalar expressions, variadic functions operator, updated *use* operator and more

Encoding for PHP 5.6 is fully supporting in version 10 of SourceGuardian. PHP 5.6 introduced new language features and updated bytecode format to support them. Files encoded with SourceGuardian 8, 9.x or older will need to be re-encoded with SourceGuardian 10 in order to run protected files under PHP 5.6

- New loaders for PHP 5.6, updated loaders for older versions of PHP

We updated loaders and created new ones for PHP 5.6. Loaders for the following operating systems are available:

- Windows 32-bit (VC6, VC9; VC11 PHP 5.5, 5.6)
- Windows 64-bit (VC9 PHP 5.3,5.4; VC11 PHP 5.5,5.6)
- MacOSX (universal binaries, include i386, x86_64)
- Linux (i386, x86_64)
- FreeBSD (i386, x86_64)

We update the following loaders on request. Please check [our blog](#) and the [loaders page](#) for new versions.

- OpenBSD (i386, x86_64)
- IBM PowerLinux
- HP-UX Itanium
- Linux ARM (armel)
- Linux ARM (armhf) including Raspberry Pi

We are happy to work with our customers if they need bespoke loaders for other custom operating system. Please contact support@sourceguardian.com if you are interested.

We do not include loaders for all supported operating systems to the GUI package. Please visit out [loaders page](#) if you can't find the loader for your OS in the [Copy Loaders](#) dialog.

- A closing PHP tag is not added anymore to the end of encoded files. It's not needed, Zend recommends to not use it. Omitting the closing tag 'automatically' protects against 'headers already sent' errors if any of the encoded files were accidentally opened/saved in the editor and some characters were added at the end. Omitting the closing tag does not affect execution of protected files.
- SG_LIC_PATH environment variable may be used to specify where the loader should search for a license file. See details for [GUI](#), for [command line](#)
- PHP short tags <? ?> are enabled by default. If you do not need them enabled for any reason, you may turn them off in [advanced settings in GUI](#) or by using the new [--no-short-tags option in the command line](#). The old --short-tags option has been removed.
- License generation fixed in GUI when a URL is specified as a license file name.
- Fixed how encoding only of changed files works in GUI.
- A new 'Refresh' button added to the GUI for updating the project tree. It is useful if any of the files were changed or added to the folders behind the encoder GUI. Also automatic refresh happens when you encode only files updated since the last encoding (see [Advanced settings](#)).
- A new --keep-file-date [command line option](#) added to keep the modification date for encoded files the same as the modification date of source files. The same option is available in GUI in [Advanced settings](#).

9.2 Version 9.5 / July 2013

Version 9.5 introduces encoding for PHP 5.5 and adds some new options. As usual this update is partly based on comments and suggestions of our users. We were glad to receive comments and suggestions and want to thank you very much for sharing your ideas! We are looking forward to hearing about other suggestions for improving SourceGuardian. Here is a list of recent version 9.5 changes.

NEW FEATURES

- Full support of PHP 5.5 encoding including the latest language options: generators, coroutines, 'finally' operator and more

Encoding for PHP 5.5 is fully supporting in version 9.5 of SourceGuardian. PHP 5.5 introduced new language features and updated bytecode format to support them. Files encoded with SourceGuardian 8, 9.0 or older will need to be re-encoded with SourceGuardian 9.5 in order to run protected files under PHP 5.5

- New loaders for PHP 5.5, updated loaders for older versions of PHP

We updated loaders and created new ones for PHP 5.5. Loaders for the following operating systems are available:

Windows 32-bit
Windows 64-bit (PHP 5.3,5.4,5.5)

MacOSX (universal binaries, include i386, x86_64)
Linux (i386, x86_64)
FreeBSD (i386, x86_64)
OpenBSD (i386, x86_64)
IBM PowerLinux
HP-UX Itanium
Linux ARM (armel)
Linux ARM (armhf) including Raspberry Pi

We are happy to work with our customers if they need bespoke loaders for other custom operating system. Please contact support@sourceguardian.com if you are interested.

We do not include loaders for all supported operating systems to the GUI package. Please visit our [loaders page](#) if you can't find the loader for your OS in the [Copy Loaders](#) dialog.

- Standard input and output support for the [command line encoder](#).
- New option to disable additional CRC check to enable eval()'ing protected code. See details for [GUI](#), for [command line](#)
- Added "Script will expire in (days)" option to the [Locking page](#).
- Changed how 'modification date' works in GUI
- The 'Copy unencoded' mode is now assigned to empty files when adding them to the project. This is to eliminate 'empty file - skipped' error messages from the encoder. It replicates what GUI does with other files that are not detected by their file extensions. If you don't need to copy empty files to the target folder, you may change the encoding mode to 'Skip' for them.
- Reverting the project was affecting modification dates - fixed
- Updated built-in support and automatic update in GUI.

9.3 Version 9.0 / July 2012

Version 9.0 introduces encoding for PHP 5.4, fixes problems and adds some new options. The update is partly based on comments and suggestions of our users. We were glad to receive comments and suggestions and want to thank you very much for sharing your ideas. We are looking forward to hearing about other suggestions for improving SourceGuardian and we are open to new ideas. Here is a list of recent version 9.0 changes.

NEW FEATURES

- Full support of encoding for PHP 5.4

Encoding for PHP 5.4 is fully supporting in version 9 of SourceGuardian. PHP 5.4 introduced new language features and updated bytecode format to support them. Files encoded with version 8 or older of SourceGuardian will need to be re-encoded with SourceGuardian 9 in order to run protected files in PHP

5.4

- New loaders for PHP 5.4, updated loaders for older versions of PHP

We updated loaders and created new ones for PHP 5.4. Loaders for the following operating systems are available:

- Windows 32-bit
- Windows 64-bit (PHP 5.3,5.4)
- MacOSX (universal binaries, include i386, ppc, x86_64, ppc64)
- Linux (i386, x86_64)
- FreeBSD (i386, x86_64)
- OpenBSD 5.0,5.1 (i386, x86_64)

We may try to compile bespoke loader for other custom operating system. Please contact support@sourceguardian.com if you are interested.

- New option to stop encoding on E_DEPRECATED PHP 5.3+ compiler errors. See details for [GUI](#), for [command line](#)
- New option to encode only changed files detected by file modification date. See details for [GUI](#), for [command line](#)
- Allow absolute file system path to a custom license file or an URL. [See details](#)
- Improved compatibility for scripts encoded for PHP 5.2.x
- Automatic filtering UTF-8 BOM from all source files. This allows encoding files created in editors that save BOM.
- Protected scripts starter code has been reworked and improved. The default starter code now recognizes "thread safety" option providing a correct loader name for systems running thread safe PHP. It also displays instructions to a user about what loader is needed, where to download it and how to install it. The old version was displaying the loader name and the error message. It is still possible to exclude the default starter code from protected scripts (using -n command line encoder option or an appropriate option in GUI). Also it is still possible to replace the standard error message about a missing loader (using -j command line option or an appropriate option in GUI) with a custom one which may be either html/text or php code. If it's PHP code a new \$__ixedurl variable may be used, it contains the required loader download link.

For information: The starter code is a non-encoded portion of php code that is added by default to encoded PHP scripts. The task of the starter code is to detect if SourceGuardian loader is present and if it is not, then try to load it automatically if PHP configuration allows to do it. If automatic loading is not possible, the starter code will display a message telling a user what loader is needed, where to download it and how to install it in order to run SourceGuardian protected files.

- Now it is possible to send licgen's output to console instead of a file, use -- (double dash) instead of the output file name. [See details](#)

- Custom constants substitution in the custom header code. All user {constants} will be replaced in the prepend code. Also some standard SG constants may be used:

{SG_DATE} - current date i.e. date of encoding

{SG_LICENSEE} - SG license owner from the SG license file

It works in the same way also for licgen and do replacements for custom text if it is used.

BUGFIXES

- Fixed displaying strict errors (--strict-errors) that did not work if displaying deprecated errors (--deprec-errors) was not specified.
- On multiple compiling error the encoder was displaying the last one, while PHP displays the first one. Fixed.
- Updated protected scripts header. A protected script will not stop execution in case of 'extension_dir does not exists' error, the script will continue and fails in a standard way or an assigned error handler function will be called. This allows to catch errors like that in the custom error handler.
- Fixed encoding of PHP predefined constants like DIRECTORY_SEPARATOR in PHP 5.3+ (re-encoding is required). Protected scripts always do dynamic lookup as values of standard constants may differ on systems depending on OS, CPU etc.
- Fixed running domain locked scripts on web servers returning a port number in HTTP_HOST/ SERVER_NAME (e.g. mydomain.com:88)
- Fixed APC and other PHP bytecode cache compatibility issue in loaders.
- Now it is possible to catch SourceGuardian errors like "SourceGuardian Loader - script header is broken [10]" etc with a standard PHP error handling mechanism.
- Fixed --short-tags option which was always on in the previous versions of the encoder.

Mac OS X GUI

- Two Project Window modes: "standard" and "file manager". This may be changed in [Preferences](#), please read further details in Preferences / Interface & Updates.
- More informative log. Added total files counter, copied files counter and list, additional warning if the encoder has not completed encoding successfully.
- Exclusions in Preferences was changed to file masks instead of file extensions.
- Project window. "Don't encode" type renamed to "Copy unencoded", added new "Skip" type which are more straightforward.
- Added "Check date with online time servers" option in [Locking options](#). This option must be selected if you have time servers specified and want your protected scripts that have an expiry date also set to check date with online time servers. This new option is off by default and you need to select it when loading an old version 8 SourceGuardian project file that has time servers specified and want to check date with online time servers.
- Support of the new options of version 9 encoder (PHP 5.4 encoding, new loaders, new advanced options etc)

NEW GUI FOR WINDOWS AND LINUX

- We are proud to present a fully reworked new cross platform GUI for Windows and Linux. It includes all the features of GUI for OSX and even more. It has built-in support system and more.

SUPPORTED PHP VERSIONS

- Encoding for PHP 4.3.x to PHP 5.4.x are fully supported

SUPPORTED OS

- Encoder is available for Windows, MacOSX, Linux (i386 and x86_64 versions).
- GUI and command line encoders and tools are included.

9.4 Version 8.2 / April 2010

Version 8.2 is partly based on comments and suggestions of our users. We were glad to receive them and want to thank you very much for sharing your ideas. We are looking forward to hearing about other suggestions for improving SourceGuardian and we are open to new ideas. Here is a list of recent version 8.2 changes.

- **Added a new predefined loader constant 'loader_version'**

`sg_get_const('loader_version')` returns a loader version number when called from the protected code.

- **Added a new license generator option to add custom text into the license file.**

We have added an option to include custom text into license files generated by SourceGuardian license generator. The included text is protected with a checksum against modification. The option may be used to include user information, license description etc into license files.

- **API function `sg_get_const()` issue fixed.**

In previous versions it was possible to get an encoded constant value from the unencoded code included from the protected one. It has been fixed.

- **The last catch block issue fixed.**

The last catch block did not terminate the execution of the code in previous versions. If you are experiencing this problem you need to re-encode your code with version 8.2 of the encoder and install the updated loader.

- **Mac GUI improvements.**

We have added an exclusion list to Preferences which lets exclude files or folders that should not be included into the project when you add files. You can easily exclude .svn folders for example.

Added an option to encode only selected folders or files - you need to press Command key while clicking the Encode button. This option is useful if you need to re-encode only some files without doing it for the entire project.

Added Custom license text in Advanced settings which reflects a new generator option mentioned above.

New Preferences screen.

Other minor fixes.

- **Documentation update.**

The loaders installation section has been revised, added important information for Windows users. Other minor changes have been done reflecting the improvements mentioned above.

9.5 Version 8.1 / January 2010

Version 8.1 is an update based mainly on comments and suggestions of our users. We were glad to receive them and want to thank you very much for sharing your ideas. We are looking forward to hearing about other suggestions for improving SourceGuardian and we are open to new ideas. Here is a list of recent version 8.1 changes.

- **New loader API functions for encryption and decryption of a string.**

We have added new loader API functions `sg_encode_string()` and `sg_decode_string()` which can be used for encryption and decryption of a string when called from the protected PHP code. See [SourceGuardian loader API](#) section for details.

- **New algorithm for searching for the loader**

We have changed the way protected scripts search for loaders when dynamic loading is possible. Now protected scripts use the following rules:

<u>Version</u>	<u>Logic</u>
PHP 5.2.5+	Loader is loaded? If no, try <code>extension_dir</code> , then custom error or standard message.
<PHP 5.2.5	Loader is loaded? If no, try to <code>dl()</code> from the current dir then parents, if not found - try <code>extension_dir</code> . If still not found, then custom error or standard message.

Note: dynamic loading is only possible from `extension_dir` since PHP 5.2.5 and we do not try any other directories for PHP 5.2.5+

- **Option to stop encoding at a critical error**

We have added a new option for the encoder to stop at a critical error. See [GUI Manual / Project / Advanced Options](#) and [Using the command line encoder / Advanced Options](#) sections for details.

- **Option to report E_STRICT compiler errors**

E_STRICT "Strict Standards" warnings were introduced in PHP 5. Now the encoder has an option to warn of such messages during encoding. See [GUI Manual / Project / Advanced Options](#) and [Using the command line encoder / Advanced Options](#) sections for details.

- **User files encoding API changes**

We have changed the way how script keys based on Project ID or Project Key (if an external license file is used) are used for encoding or decoding files with `sg_load_file()` and `sg_encode_file()` loader API functions. Now we keep track of which protected PHP script uses which key and the loader will use an appropriate key for encryption/decryption of user files and strings (see new `sg_encode_string()` and `sg_decode_string()` API functions). Older versions of loaders would use the key of the last loaded protected script. These changes allow to use the encoding functions correctly in a situation when one protected script is inherited from another protected script and the above encoding/decoding functions are called. See [Creating custom encoded files from protected scripts](#) section for details of using these functions.

- **Marking a file to be skipped during encoding**

It's possible to mark a file to be skipped by the encoder. Add the following string anywhere in the code, use comments. Skipped files will be copied as-is to the target folder if it's specified.

SourceGuardian:DO_NOT_ENCODE

E.g. `/* SourceGuardian:DO_NOT_ENCODE */`

Note: Comparison is case sensitive for Windows and Solaris if you will ever move to any of these platforms for encoding your source files. Do not change the case for better code compatibility.

- **IPv4 address mapped to IPv6 is correctly recognized**

The loaders now correctly understand IPv4 mapped to IPv6 `::ffff:XX.XX.XX.XX` style IP addresses in `$_SERVER['SERVER_ADDR']`. This fixes issues with locking to IP when a webserver returns IPv6 style address.

Note: SourceGuardian does not support locking to IPv6 yet, only to IPv4 addresses.

- **Tested compatibility with xdebug extension**

We have tested compatibility of loaders and xdebug PHP debugging extension. There is NO any compatibility issue except for the scripts encoded with `--conj` (script will only work with other encoded files) option. As xdebug modifies the bytecode files encoded with this option will fail with a conjunction error (code [15]). It's NORMAL behaviour demonstrating one of the points of protection. Obviously the internals of the encoded script cannot be debugged so once the debugger reaches the `sg_load()` function the entire protected script will be executed in one step.

- **Tested compatibility with APC**

We have tested compatibility of loaders and APC PHP cache extension. APC v3.0.19 the latest stable was tested. All protected scripts get cached no problem.

- **Automatic update of the project's contents for Mac OS X GUI**

The contents of a project (files and folders added to the project) will be automatically synchronized with the contents of real folders on a disk when the project is reopened in SourceGuardian. If there are any changes within the added folders then the inner list of files and folders will be recursively updated. If there are any changes of the folders or files which were directly added on the top level then the application will query whether to remove missed elements or leave them within the project.

Note: Newly added files will get an encoding type according to their file extensions and settings in preferences. Probably you will want to check the type setting for new files.